Master Thesis
Electrical Engineering
Thesis no: MEEOH27
November 2008

A Study of Fast Backprojection Algorithm for UWB SAR and a Comparison between Fast- and Global Backprojection

Hawa Yves

School of Signal Processing
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden.

This thesis is submitted to the Department of Signal Processing, School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering with emphasis on Telecommunication and signal processing. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:
Author(s): Hawa, Yves
Address: Bremergatan 25
39 233 Kalmar
Sweden
E-mail: yveshawa@gmail.com

University advisor(s):
Viet T. Vu
E-mail: vtv@bth.se
School of Engineering
Blekinge Institute of Technology

University advisor(s):
Thomas Sjögren
E-mail: tsj@bth.se
School of Engineering
Blekinge Institute of Technology

School of Engineering
Blekinge Institute of Technology
Box 520
SE-372 25 Ronneby
Sweden.

Internet: www.bth.se/tek
Phone: +46 457 38 50 00
Fax: +46 457 271 29

Abstract

In this thesis, we present a study of the Fast backprojection (FBP) known to be a fast time-domain algorithm for image retrieval in Synthetic Aperture Radar (SAR). As a time-domain algorithm, FBP possesses inherent advantages such as perfect motion compensation, unlimited scene size, wide bandwidth and ability to handle long integration angles. Although FBP reproduces SAR images on pixel-by-pixel basis, the processing time for FBP is reduced significant compared to the Global Backprojection (GBP) with a heavy computational load. For GBP, the number of operations to process a $N \times N$ SAR image with N aperture positions is proportional to N^3. Whereas, the number of operation required by FBP in the same case is reduced by a factor of $\sqrt{N}$ i.e. proportional to $N^2\sqrt{N}$. We give a detailed explanation on how to implement FBP in Matlab in order to retrieve a SAR image. We explain about different techniques of interpolation used to obtain a high quality image. We also present a method to compare between interpolation techniques in term of SAR image quality and processing time. A comparison between FBP and GBP is also a topic to be discussed in this thesis. In this comparison, the processing time in theory and reality will be focused.

Keywords*:* Synthetic Aperture Radar, Ultra Wide-band, time-domain algorithms, Global Backprojection, Fast Backprojection, SAR image processing, interpolation techniques.

Acknowledgements

The author is grateful to his supervisors Mr. Viet T. Vu and Mr. Thomas Sjögren for their interest in this work and for their insightful comments and suggestion and specially by their professional guidance that assisted the author in improving the presentation of this Thesis. I would like to express my sincere thanks to Dr. Jörgen Nordberg for accepting me and giving me an opportunity to work in this field, to my family who gave me the support all over these years, to my friends who supported me during my work by discussing and reviewing. Last but not least, I would like to thank Sweden for giving us an opportunity to get a high-quality education.

Yves Hawa
November 05, 2008
Ronneby, Sweden.

Table of Contents

List of Figures

List of Tables

Chapter 1: INTRODUCTION

RADAR is an acronym for ‘‘Radio Detection and Ranging’’. It is a system that uses electromagnetic waves to identify the range, altitude, or speed of both moving and fixed objects. Radar is an important technology due to its all-weather, day night capability of detecting, locating and imaging. It consists of a transmitter that emits radio waves propagated at the speed of light, which are reflected by the target and detected by a receiver placed normally at the same location of the transmitter. Knowing the speed of the signal, the range is calculated by measuring the time it takes for the signal to travel from the transceiver back to the receiver. Radars can be either fixed or moving. Airborne or spaceborne radar is an example of moving radars. Airborne radar is equipment based on aircraft platform that is used for several purposes such as weather assessment, navigation, mapping, and military combat [1].

Synthetic Aperture Radar (SAR) is a form used on airborne radar. An airborne SAR is radar equipment based on aircraft platform to produce high-resolution mapping of earth surface in both range and azimuth dimensions. SAR have been in the market since 1950, but due to the lack of computer power and advanced digital signal processing algorithms the SAR system could not be used in an efficient way [2]. SAR allows high-resolution imaging in two dimensions from low-resolution aperture data where the image is mapped from the received signal energy. SAR has an improved resolution in both range and azimuth than normal radars. Like all the radars systems, narrow pulses yield to a fine range resolution. To get a fine azimuth resolution, a physically large antenna is needed. This is an impossible task to implement in reality. SAR can be seen as an alternative way to get high azimuth resolution. SAR collects the data while flying and then processes it as it comes from a physically long antenna. In other words, the radar moves along the flight track where it sends pulses and receives echoes. The distance the aircraft flies in synthesizing the antenna is known as the synthetic aperture [3].

There have been several image formation algorithms which are basically divided into two major groups: Time domain algorithm and Frequency domain algorithm [4]. Due to computational efficiency, frequency-domain algorithms are mainly used. The shortcoming is that they are derived with assumption of linear aperture or flight track. However, this assumption is not totally valid for an ultra-wideband system (UWB). The time domain backprojection schemes circumvent the motion compensation problem but require very heavy computational load [2], for

example, Global Backprojection (GBP) algorithm which is known as a basic time domain algorithm. Later, other time domain algorithms like Fast Backprojection (FBP) and Local Backprojection (LBP) have been proposed with faster processing time but still retain advantages of GBP [4]. Multiple researches have been implemented to compare these algorithms together for different purposes. In [4], a comparison between a time domain algorithm and frequency domain algorithms with the focus on the SAR image quality measurements based on the Integrated Sidelobe Ratio (ISLR) and Peak Sidelobe Ratio (PSLR) and processing time.

Interpolation technique is an important task in SAR image processing integrated into algorithms. In GBP, the range interpolation of SAR data is performed using a suitable technique in term of image quality required, whereas, no interpolation is needed in azimuth. In FBP, the radar data is transformed to a polar coordinate that is selected to be a more optimum case for the interpolation since the polar grid is thin in the angular direction. The quality of SAR image depends on the interpolation techniques used throughout the processing. These approximations affect the final image quality which will be another major problem for most frequency domain algorithms because they require interpolation of data in the frequency domain which will cause artifacts that will be spread over the entire image [5]. A more rigorous interpolation algorithm will involve more operations and therefore needs to be traded for the image quality against processing time [6]. Multiple researches have been implemented to compare these interpolation techniques together for different purposes. In [7], a comparative study based on the difference in the interpolation techniques between stages and the sidelobes in azimuth is presented.

In this thesis, we focus on the time domain algorithms especially the Fast Backprojection (FBP) trying to proof the improvement in the computational load, the image quality and the processing time of the FBP with respect to the Global backprojection (GBP). The SAR system which is simulated is an ultra-wideband system which operates in the low VHF-band (20-90 MHz). We begin by studying the FBP algorithm, implementing it in Matlab and comparing it to the GBP and then evaluating the improvement with different interpolation techniques.

Chapter 2: SYNTHETIC APERTURE RADAR

This chapter introduces the baseline of this master thesis to provide the reader with an overview of the whole paper and the important issues to be discussed and investigated. Before we begin addressing implementation of the Fast backprojection (FBP), we provide an overview of all the basics referenced in this thesis including radars, Synthetic Aperture Radar (SAR), chirps signals, matched filter, and different interpolation techniques.

2.1 Radar and Synthetic Aperture Radar

In this section, we explain the basic concepts of radar operation, equation and pulse compression, as well as giving an introduction to the Synthetic Aperture Radar, its functionality, its algorithms and different interpolation.

2.1.1 Radar

Radar, as mentioned in the introduction chapter, is a system for radio detection and ranging that uses electromagnetic radio waves to detect targets and determine the target range [1]. The system works by sending a pulse and parts of that signal will be reflected back to the radar antenna after it reaches a target with a constant speed close to the speed of light. Radar is used in many contexts, including meteorological detection of precipitation, measuring ocean surface waves, air traffic control, police detection of speeding traffic, and for military purposes [3].

It is a system that sends and receives electromagnetic waves to identify the range and speed of a target. The amount of power P_r returning to the receiving antenna is given by the radar equation [8]:

$$P_r = \frac{P_t G_t A_r \sigma F^4}{(4\pi)^2 R_t^2 R_r^2} \tag{2.1}$$

Where P_t is the transmitter power, G_t is the gain of the transmitting antenna, A_r is the effective aperture (area) of the receiving antenna, σ is the radar cross section of the target, F is the pattern propagation factor, R_t is the distance from the transmitter to the target, R_r is the distance from

the target to the receiver. Thus, the longer the target is illuminated, the more energy will be reflected back to the receiver. In the usual case, whereby the transmitter and the receiver are at the same location, $R_t = R_r$ and the term $R_t^2 R_r^2$ can be replaced by R^4, where R is the range, thus the equation of the radar is reduced to the expression 2.2:

$$P_r = \frac{P_t G_t A_r \sigma F^4}{(4\pi)^2 R^4} \tag{2.2}$$

This shows that the received power declines as the fourth power of the range, meaning that the reflected power from distant targets is very small. This equation can be simplified further if we consider that $F = 1$, which represents vacuum without interference [8], or means that the antenna is isotropic or omni-directional.

2.1.2 Synthetic Aperture Radar

Synthetic Aperture Radar is a type of radar mounted on an aircraft or a satellite that can take high-resolution radar imaging in two dimensions from low-resolution aperture data. It can be used over relatively immobile targets and moving targets, as we will discuss later in this chapter. SAR is a radar antenna that is attached to the side of an aircraft. A single pulse from the antenna will be broadcasted and will illuminate the terrain from directly beneath the aircraft, out to the horizon. Thus, if the terrain is approximately flat, the time at which echoes return allows points at different distances from the flight track to be distinguished. Distinguishing points along the track of the aircraft is difficult with a small antenna. However, if the amplitude and phase of the signal returning from the ground are recorded, and if the aircraft emits a series of pulses as it travels, then the results from these pulses can be combined. Effectively, this series of observations can be combined just as if they had all been made simultaneously from a very large antenna; this process creates a synthetic aperture much larger than the length of the antenna [3]. So if the antenna is long enough, the resolution will be perfect. However, building a long antenna is an impossible task. Alternatively, if the radar is moved along a one kilometre synthetic straight line where it sends and receives echoes along the whole synthetic line, one would have a one kilometre synthetic radar [3]. Now, combining this series of observations requires significant computational resources. It is normally done at a ground station after the observation is complete, using different types of algorithms. But, when storing the echo, the exact position of where the echo comes from is unknown; only the range and the lobe width are identifiable. This is done for every pulse, together with digital signal processing algorithms that will later be processed to

obtain a high resolution image [8]. The result is a map of radar reflectivity including both amplitude and phase. The amplitude and phase contain information about ground cover, where the phase information is more important than the amplitude information.

2.1.2.1 Modes of SAR

The Synthetic Aperture Radar has three different operating modes, the stripmap, scan, and spotlight as shown in Figure 2.1 [1].

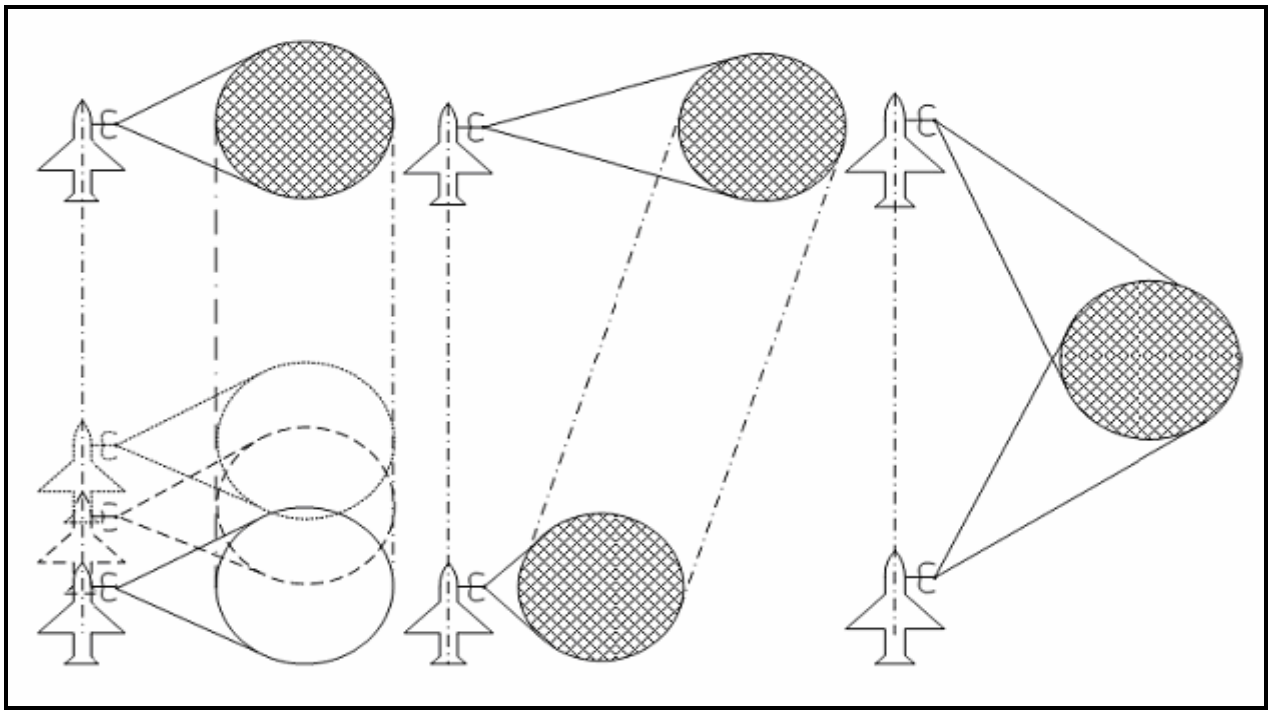

Figure 2. 1: Three different SAR methods: Stripmap, scan and spotlight [1].

During a spotlight mode data collection, the sensor steers its antenna beam to continuously illuminate the terrain patch being imaged. In the stripmap mode, the antenna pointing is fixed in relation to the flight line. This results in a moving antenna that sweeps along a strip of terrain parallel to the path of motion, as we can see in the figure above. In the scan mode, the sensor steers the antenna beam to illuminate a strip of terrain at any angle to the path of motion [1].

Spotlight mode gives a better resolution than that offered by the stripmap mode, using the same physical antenna. It also offers the possibility of imaging a scene at multiple viewing angles during a single pass, and it allows efficient imaging of multiple smaller scenes where the stripmap mode images have a long strip of terrain [1]. The SAR azimuth resolution improves with the length of the synthetic aperture. Therefore, the azimuth resolution of the physical

aperture decreases according to the distance, but the synthetic apertures resolution can be made irrespectively of distance because of the property of SAR. That means a resolution below one metre can be achieved by airplanes and satellites from a large distance. In other words, the stripmap mode's azimuth antenna beam width limits the available synthetic aperture length, but in the spotlight mode this is not the case. Since we are using a very wide beam, the mode used in this thesis work is a combination between the stripmap and the spotlight.

Normal radar emits pulses with a very narrow range of frequencies. This places a lower limit on the pulse length and consequently on the resolution in the distance direction. Additionally, the interpretation of the results is eased by the fact that the material response must be known only in a narrow range of frequencies. However, the Ultra-Wide Band radar (UWB) emits very short pulses consisting of an extensive range of frequencies. Such pulses allow high distance resolution but the information is concentrated in relatively low frequencies with long wavelengths. Thus such systems require very large receiving apertures to obtain high resolution corresponding along the track. Due to the fact that the information is captured in low frequencies, the most relevant material properties are those at lower frequencies, which are different for most radar systems. In particular, this type of radar system has the capability of penetrating more distance into soil [8].

As a characteristic, the Synthetic Aperture Radar is a system that is capable of detecting moving targets. Maybe a new invention relating to a radar system capable of detecting moving targets is presented in [9], which includes a platform that moves over a number of objects and supports radar equipment which reproduces the objects by means of a Fast backprojection synthetic aperture technique. The system has no requirement for directivity or fractional bandwidth. It is therefore possible to use it with an Ultra Wide Band (UWB) SAR system and Wide Beam (WB) transmission and reception. Thus the UWB-WB SAR at low frequencies will add the capability of detecting targets moving in forested areas, and at microwave frequencies, it will provide the capability of high resolution images of the moving target [9].

2.1.2.2 Resolution obtained by SAR

Having given an introduction and a detailed description of the SAR in the previous paragraph, we now propound an overview of how the Synthetic Aperture Radar imaging concept works. As shown in the figure 2.2 [3], the airborne SAR imaging is perpendicular to the aircraft's velocity. As mentioned in the introduction of this chapter, SAR creates a two dimensional (2D) image.

The first dimension is called the range. This is a measure of the line of sight distance from the radar to the target. That is why both resolution and measurement are realised in SAR in the same way as most other radars. It is also possible to calculate the range by measuring the time from the transmission of a pulse, and the time to receive the echo from the target. The range resolution is calculated by a transmitted pulse width, i.e. narrow pulses yield fine range resolution [3].

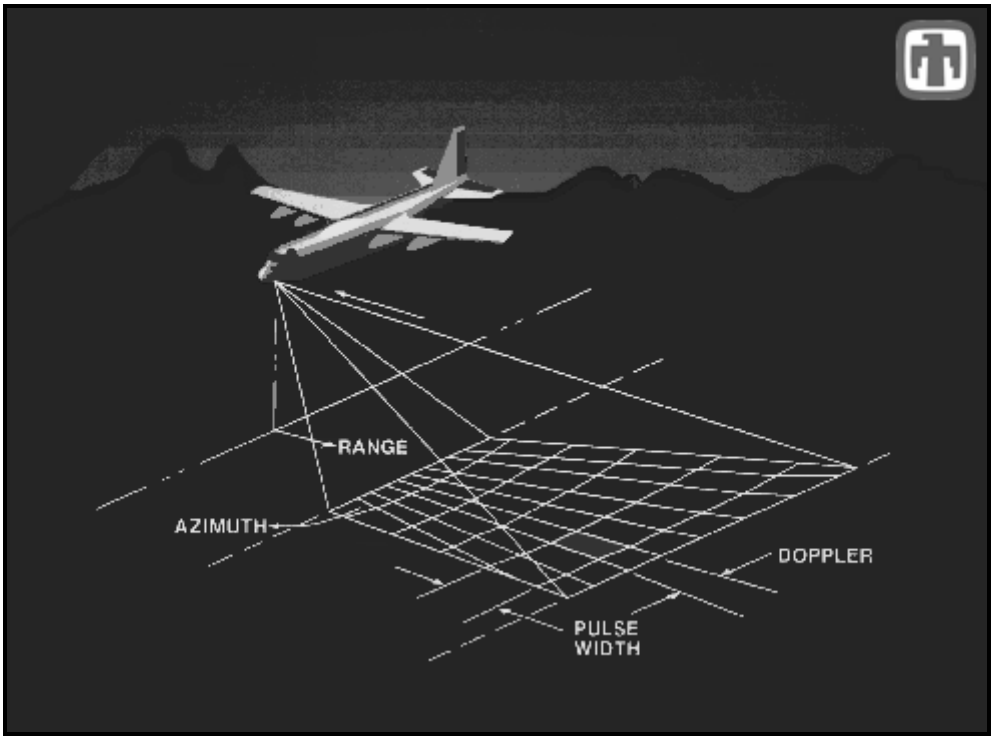

Figure 2. 2: Synthetic Aperture Radar Imaging Concept [3].

The second dimension is known as azimuth. This dimension is perpendicular to range. SAR can produce azimuth with a high standard resolution, making it different from other radars. In order to get this kind of resolution, a physically large antenna is required, to focus both transmitted and received energy into a sharp beam. The azimuth resolution is defined by the sharpness of the beam. The same concept is used in optical systems, for example telescopes, which require large aperture like mirrors or lenses that are analogous to the antenna of the radar to get fine imaging resolution [3].

A narrow synthetic beam width yields to a better resolution. The position of a target calculates the Doppler frequency of its echoes. Therefore the target ahead of the aircraft realises a positive Doppler offset, whereas the target located behind the aircraft produces a negative offset. The aircraft flies a distance which is called synthetic aperture. The azimuth is determined by echoes

which are resolved into a number of Doppler frequencies. However, it is generally not practical to transmit short pulses for providing range resolution. Usually the antenna transmits longer pulses with wide bandwidth modulation. This has the effect of complicating the range processing, but decreases the peak power requirements on the transmitter. It is possible to moderate azimuth resolutions, and indeed a target's range to each location on the synthetic aperture changes along the synthetic aperture. The target reflects energy which must be mathematically focused, in order to compensate for the range dependence across the aperture prior to image formation. Thus, for fine resolution systems, both range and azimuth processing are coupled, and dependant on each other. As a consequence, this has a large increase in the computational processing [3].

As previously mentioned, a radar map must provide high resolution in both range and azimuth dimensions. High range resolution can be achieved by pulse compression techniques as well as radar systems. High resolution cannot normally be achieved by conventional operations; however, SAR can provide a relatively high resolution in azimuth compared to other radars, which is an important advantage to other radars. The azimuth resolution Δ_a for an antenna of SAR can be calculated by [1].

$$\Delta_a = R \cdot \frac{\lambda}{D} \tag{2.3}$$

R is the distance from antenna to target, D is the physical dimension of antenna, and λ is the wave length. From the formula 2.3, we can see that if we want to obtain a higher resolution in azimuth, the length of antenna must be increased. Consequently, if the target distant R is equal to one kilometre, wave length λ = 50 cm, and users require azimuth resolution Δ_a = 50 cm, then antenna length would be one kilometre, which is impossible in practice. By considering this impossibility, instead of building a large physical phased array antenna, a single array element that moves through successive element positions to form the complete array would be a good choice [1]. Since waves travel at the speed of light, we could neglect the speed of aircraft by using start-stop-approximation in this case. Along this route, the single array element sends and receives echoes at each position. Thus the data collected from each position is coherently combined to simulate a large array antenna, usually in microwave hardware. We then put the range bins together to create an echo matrix used by digital processor for further processing.

Since we make the assumption of start-stop-approximation, the plane would remain still during two element-positions. Then there would be no Doppler Effect in the process as mentioned previously in this chapter. However, this is not true in the real situation; the Doppler Effect exit and echoes have different frequency shift. To reduce this error, we could calculate the frequency shift by using distance between two echoes.

2.1.2.3 Chirp Signals

A chirp signal is a way of handling a practical problem in echo location systems like radar and sonar. It has also other applications, such as in spread spectrum communications. A chirp is defined as a signal in which the frequency increases or decreases with time. Figure 2.3 shows the plot of the chirp signal. The equation 2.4 is used to create the chirp signal where τ is a vector of time from $[-t_p/2 \;\; t_p/2]$ where t_p is the pulse duration, f_c is the center frequency and k is the chirp rate. The signal frequency will increase when τ increases.

$$S_{chirp}(\tau) = e^{j2\pi\left(f_c\tau + \frac{k\tau^2}{2}\right)} \tag{2.4}$$

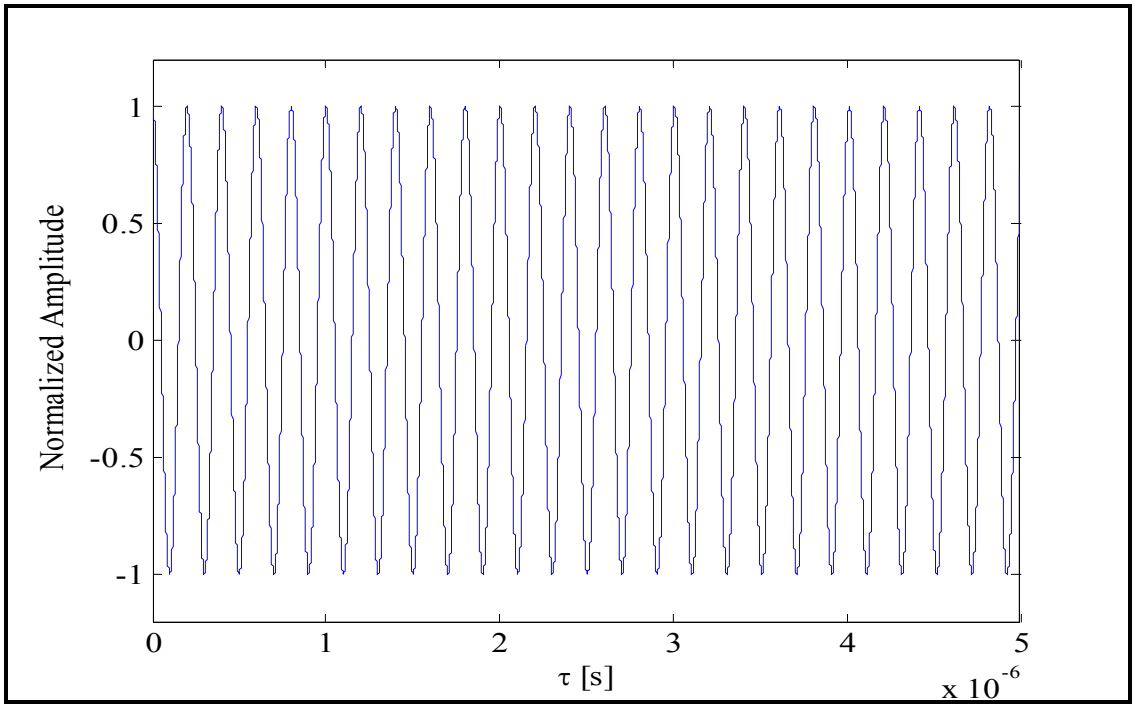

Figure 2. 3: Plot of the chirp signal.

The key characteristic for the chirp system is that it is completely reversible. It is possible to turn the signal into an impulse if you run the chirp signal through an anti-chirp system. In order to do that, the anti-chirp system needs to have a magnitude of one [10].

The pulse is compressed in time, so there is an effect of a much shorter pulse, which also makes a better resolution while having the benefit of longer pulse length. There is a second requirement in which we need to detect objects farther away. To do this, more energy is required in the pulse. However, there is a conflict requirement between more energy and shorter pulse. To provide the pulse, we need an electrical power. This is equal to the energy of the pulse, divided by the length of the pulse. As it requires more energy and a shorter pulse, the electrical power can handle a limiting factor in the system. Therefore, a way of breaking this limitation is provided by chirpsignals. It passes through a chirp system before the impulse is able to reach the final stage of the radio transmitter. A chirp signal is used, instead of bouncing an impulse off the target aircraft. The signal passes through the anti-chirp system in order to restore the signal to an impulse, only after the chirp echo is received. After this, short pulses are visible through the portions of the system that measure distance, while long duration signals are seen by the power handling circuits. This kind of wave shaping is very important, as a fundamental part of modern radar systems [1].

2.2 SAR Processing Algorithms

As explained in the introduction chapter, the early SAR systems mostly use the frequency domain techniques due to their computational efficiency. There is, however, a major drawback. The algorithms are only valid for the linear case when flight trajectory is straight and the speed is constant. The algorithms are still usable, since the flying trajectory can be considered linear in short distances. Range Migration algorithm (RMA) which is also known under another name $\omega - k$ algorithm, Range Doppler (RDA) and Chirp Scaling (CSA) are examples of frequency domain algorithms. RMA and RDA algorithms require computationally expensive interpolation. CSA algorithm avoids interpolation and can simply be employed through complex multiplications and Fast Fourier Transforms (FFT). This can be seen as an advantage of CSA algorithms compared to other frequency-domain algorithms [4]. One way of solving the flight trajectory linearity is to do the calculations in the time domain. For the SAR case, the exact solution is called Global Backprojection (GBP). This has, however, traditionally been considered

very computationally demanding, but there are a couple of different algorithms available to reduce the complexity of the calculations. These algorithms fall under the name Fast Backprojection (FBP) algorithms. Several time-domain algorithms were proposed, for example the Local Backprojection (LBP) and Fast backprojection (FBP) with the number of operations needed proportional to $N^2\sqrt{N}$ where N is the number of aperture positions or Fast Factorized Back-projection (FFBP) developed from LBP running roughly $N / \log(N)$ times faster than GBP, while still retaining the advantages of GBP [4]. These are several types of SAR algorithms used. In this thesis, we will focus on the time domain algorithms, and specifically on the Fast backprojection (FBP). These are to be examined in detail in the subsequent chapters.

2.3 Interpolation Techniques

Interpolation is a method for constructing new data points within the range of a discrete set of data points that we know. This is the definition in the mathematical subfield of numerical analysis. For another field, engineering and science, one often has a number of data points, which have been obtained by sampling or experiment. Subsequently, one attempts to build a function which will closely fit those data points. This is called 'curve fitting' or 'regression analysis' [10].

Interpolation is an example of specific curve fitting. The function must go exactly through the data points. The approximation of a complicated function by a simple function incurring a different problem is closely related to the interpolation. For example, if we suppose that we know the function, but it is too complex for us to evaluate efficiently, then it is possible to pick a few data points which can be obtained from the complicated function, and create a lookup table. Using this we can try to interpolate those data points to construct a simpler function.
It is certain that when we use a simple function for calculating new data points, we do not usually receive the same result that was achieved when using the original function. However, depending on two things, the problem domain and the interpolation method used, the gain in simplicity may offset the error [10].

2.3.1 Linear interpolation

Linear interpolation is a method of curve fitting using linear polynomials. Advantages of this method are its simplicity and straightforward employment, with relatively satisfying results.

Linear interpolation consists of simply connecting sampling points with straight lines. It is heavily employed in mathematics, particularly in numerical analysis and numerous applications including computer graphics. It is the simplest of all forms of interpolation and the most commonly used. It has the following formula in 2.5 [10]:

$$\hat{y}(n-\eta) = (1-\eta)\cdot y(n) + \eta \cdot y(n-1) \tag{2.5}$$

where η is desired fractional delay and $y(n)$ is the function we want to interpolate.

2.3.2 Linear interpolation as Convolution

A linear interpolation equivalent for filtering the continuous-time weighted impulse train is given in the equation 2.6 and 2.7 below [10]:

$$\sum_{n=-\infty}^{\infty} y(nT)\cdot\delta(t-nT) \tag{2.6}$$

with the continuous-time triangular pulse FIR filter followed by sampling at the desired phase

$$h(t) = \begin{cases} 1-\dfrac{t}{T} & t \le T \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

followed by *sampling* at the desired phase.

2.3.3 Sinc interpolation

The ideal interpolation is band-limited interpolation, i.e. the samples are uniquely interpolated based on the assumption of zero spectral energy for $|f| \ge f_s/2$. Therefore, the ideal band-limited interpolation is the sinc interpolation. Sinc interpolation is designed to minimise aliasing in a signal. The function sinc(πx) is shown in the Figure 2.4 below:

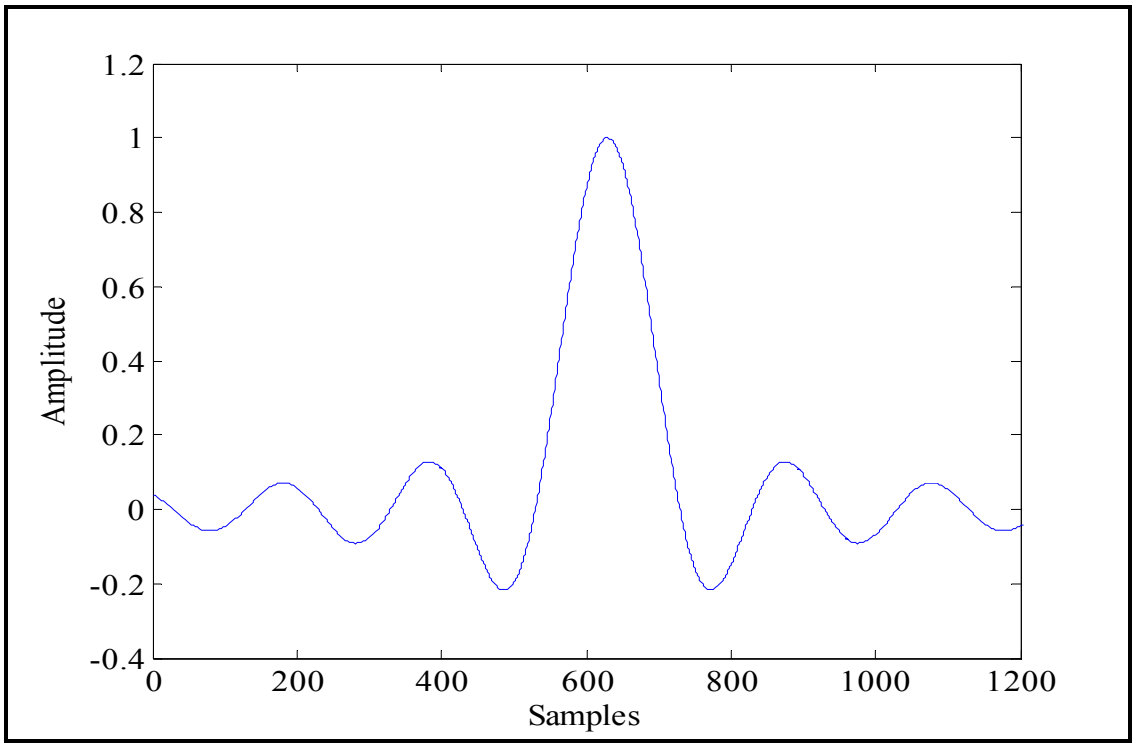

Figure 2. 4: Sinc function.

The result is that sinc(0) = 1, but its value at any other integer is zero. Each sample's contribution to the signal is a sinc function centred on the sample. The sinc function is scaled to match the height of the sample. The frequency of the sinc function is set to match the sample rate so that all neighbouring samples occur where the sinc function goes to zero, at integer values. The overall signal is the sum of all of the sinc functions of all of the samples. Sinc interpolation can be accomplished by adding together a number of sinc functions on both sides of the point being interpolated. A windowing function can improve the interpolation. A window is used to gradually fade out the sinc values as they get further away from the centre sample. This is called 'windowed sinc interpolation' [10].

Here, an explanation of a few types of interpolation. We will utilise these later in the implementation of the Fast backprojection in order to retrieve high quality SAR images. To secure a good data estimation, an advanced and precise interpolation method does not need as high a degree of oversampling as a simpler one. Hence, the total memory requirement becomes typically smaller when the interpolation becomes more advanced. However, since the computational complexity in the interpolations grows rapidly with the size of the interpolation, so the total computational requirements increase.

CHAPTER 3: FAST BACKPROJECTION

In this chapter, we explain the Fast backprojection, how it works and how we implement it in Matlab to form a SAR image. We discuss the subaperture division, the polar image formation, the transformation from polar image to Cartesian image, the computational load and the interpolation techniques. A comparison between interpolation techniques with regard to image quality is also discussed in this chapter.

3.1 Polar and Cartesian Coordinates

Since the Fast backprojection is based on two concepts polar and Cartesian image, we therefore give an overview of Polar and Cartesian coordinates and the conversion between these coordinates.

In Cartesian mathematics, each point in a plan is determined by two numbers called the *x-coordinate* and the *y-coordinate* of the point denoted as (x, y). To define the coordinates, we must use two perpendicular directed lines. Therefore, (x, y) are specified. We also need to specify the unit length, marked off on the two axes. Another use of the Cartesian coordinate system is engaged in space, where three coordinates are used in higher dimensions [11].

The Polar coordinate system (r, θ) is also a two dimensional coordinate system. Here, each point on the plan is determined by an angle and a distance. The usefulness of the polar coordinate system can be seen in situations where the relationship between two points is the most easily expressed in terms of angles and distances. In the case of the Cartesian or rectangular system, a relationship like this could only be found through trigonometric formulation. As the coordinate system is two dimensional, each point is determined by two polar coordinates which are the *radical coordinate* and the *angular* one. The radical coordinate, often denoted r and called *radius*, expresses the distance from the point to the central point that is equivalent to the origin in the Cartesian system. The angular coordinate expresses the measurement in the trigonometric way of the angle between the point and the semi-line of 0° of angle, called *polar axis* that is equivalent to *abscissa axis* in Cartesian coordinate plane [11].

A point (r, θ) in the polar coordinates can be presented in the Cartesian coordinates (x, y) by using the trigonometric functions sine and cosine:

$$\begin{aligned} x &= r \cdot \cos\theta \\ y &= r \cdot \sin\theta \end{aligned} \tag{3.1}$$

while the two Cartesian coordinates x and y can be converted to polar coordinates r by

$$r = \sqrt{x^2 + y^2} \tag{3.2}$$

This is a simple application from the Pythagorean Theorem. However, in order to calculate the angular coordinate θ, we must consider the following two ideas: Firstly for $r = 0$, we can set θ to any real value. Secondly, for $r \neq 0$, for getting a unique representation for θ, an interval of size 2π must be the limit. Therefore, we can say that the conventional choices for such intervals are $[0, 2\pi)$ and $(-\pi, \pi]$ [11].

3.2 Introduction to Fast Backprojection

Before explaining about the Fast backprojection and its implementation, we shall give an overview of how the time domain algorithm sgenerally form the image. SAR produces a high resolution map of the ground, while the platform is flying past it. In SAR the radar transmits a relatively wide beam to the ground, illuminating each resolution cell over a long period of time. Thus, while the plane is flying, the SAR system creates a map over a continuous ground. The system continuously processes incoming data and produces the image.

The ‘‘integration angle’’ is the angle between the start point and the end point of the plane where the aim point is on the middle of the flight track. The effect of the plane movement is that the distance between a point on the ground and the antenna varies over the data collection interval. This variation in distance is unique for each point in the area of interest. Therefore, when the radar transmits pulses that are repeated with a certain Pulse Repetition Frequency (PRF), M samples or range bins are collected. Each range bin corresponds to a particular distance to the ground cell, as the distance is proportional to the time until the echo is received. Sampled echoes from L radar pulses are placed in a radar data matrix. A row in the matrix corresponds to a particular pulse, and consequently to a platform position along the flight path. Likewise, a column corresponds to a particular target range. At this point the signal processing work comes.

It is responsible for integrating each resolution cell in the output image, the instantaneous response that a target in that particular cell would have [6].
The image of the size *M* x *N* pixels is created where *M* is range size and *N* is the azimuth size. Several algorithms can be used to compute this integral and the choice is a trade-off between accuracy and processing time. The images can be calculated with FFT techniques in the frequency domain or with backprojection techniques in the time domain, depending on the objectives [1]. In this thesis, we focus on the time domain algorithms.

The Fast backprojection is based on the backprojection technique to reduce processing time. It splits the whole synthetic aperture into subapertures where each subaperture forms an image. The final image is retrieved by adding all the subaperture images together [5]. Fast backprojection works with the Cartesian and polar image, thus the radar data will be transformed to the polar coordinates rather than the Cartesian coordinates, where the Global backprojection works. To transform the data to polar coordinates, we have to create a polar grid for every subaperture to form the image. The subaperture images are then formed in the polar coordinates, whereas the final image will be formed in the Cartesian coordinates. These subaperture images have a very low resolution in the angular direction. When we will combine all the subapertures into the final image, we have to upsample them to full resolution in angular direction [5]. Thus, the number of operations to perform the image is reduced by a factor of $\sqrt{N}$ comparing to the number in the Global backprojection. When we combine the entire subapertures together, we implement interpolation to obtain a reasonable quality image. Interpolation algorithms will add more operations through the processing. Therefore, there will be a trade-off between the image quality and the processing time. The interpolation in Fast backprojection when we map from polar to Cartesian image is a double interpolation in the range and azimuth direction.

3.2.1 Subaperture Processing and Nyquist Sampling Rate

Assuming we have a synthetic aperture of any shape of total length *L,* the data is backprojected by the integral shown in equation is 3.3 [5] below:

$$I(\vec{p}) = \int_0^L F\left(s, \frac{2}{\mathrm{c}} \left| \vec{p} - \vec{q}(s) \right| \right) ds \tag{3.3}$$

We divide the full aperture into N_{sub} number of subapertures of length $l = \frac{L}{N_{\text{sub}}}$. Thus, the image $I(\vec{p})$ will become a sum of subaperture images $I_{\text{n}}(\vec{p})$ as shown in the equation 3.4 [5] below:

$$I(\vec{p}) = \sum_{n=1}^{N_{\text{sub}}} I_{\text{n}}(\vec{p}) \tag{3.4}$$

where the subaperture images are given by

$$I_{\text{n}}(\vec{p}) = \int_{-l/2}^{l/2} F\left(s_{\text{n}} + \xi, \frac{2}{\text{c}} \left| \vec{p} - \vec{q}(s_{\text{n}} + \xi) \right| \right) d\xi \tag{3.5}$$

And the centre of the n-th subaperture is calculated by $s_{\text{n}} = \left(n - \frac{1}{2} \right) l$.

No approximations were made in equation 3.4. It is simply a division of the integral in equation 3.3 into the sum of a series of sub-integrals. We now concentrate on evaluating a single subaperture image in the explanation of FBP [5].

The use of polar coordinates has a big advantage in the bandwidth [5]. To conclude how closely to space the pixels in the polar grid subaperture image are located, [5] presents a derivation of simple expressions for the Nyquist rate of polar grid images. When computing the Nyquist sampling rate, they assume that the flight track is linear, which gives reasonable assumptions for the short subapertures. But during the implementation of the actual image formation, they used the non linear flight track, where they couldn't have used it in computing the Nyquist sampling rate, since the rates will be high [5]. They start by expressing the image in terms of the polar coordinates (r, α). So, the coordinate *r* is defined as distance from the centre of the subaperture as shown in the equation 3.6 [5] below:

$$r = \left| \vec{p} - \vec{q}(s_{\text{n}}) \right| \tag{3.6}$$

The coordinate *α* is defined as the cosine of the angle between the fight track and the line of sight to the pixel [1] and it is shown in the equation 3.7 [5] below:

$$\alpha = \frac{1}{r} \left[\vec{p} - \vec{q}(s_{\text{n}}) \right] \cdot \vec{q}'(s_{\text{n}}) \tag{3.7}$$

The distance between the image pixel and the point on the aperture is also expressed in the polar coordinates following the basis identity:

$$\left| \vec{a} - \vec{b} \right| = \sqrt{\vec{a}^{\,2} + \vec{b}^{\,2} - 2\vec{a}.\vec{b}}$$

The subaperture equation image in the polar coordinates is given as:

$$I_{\mathrm{n}}(r,\alpha)=\int_{-l/2}^{l/2} F\left(s_{\mathrm{n}}+\xi,\frac{2}{\mathrm{c}}\sqrt{r^2+\xi^2-2r\xi\alpha}\right)d\xi \tag{3.8}$$

Finally, we expand $F(s,t)$ in terms of its Fourier transform, and then compute the Fourier transform $\widetilde{I}_{\mathrm{n}}(k_{\mathrm{r}},k_{\alpha})$ of $I_{\mathrm{n}}(r,\alpha)$ with respect to *r* and α [5]. The bounds on $k_{r,}$ and k_{α} are then translated into Nyquist sampling requirements for *r* and α as shown in the equations 3.9 and 3.10 [5] below:

$$\Delta_{\alpha}\leq\frac{\mathrm{c}}{2f_{\max}l} \tag{3.9}$$

$$\Delta_{\mathrm{r}}\leq\frac{\mathrm{c}}{2(f_{\max}-f_{\min})} \tag{3.10}$$

where $f_{\min}$ and $f_{\max}$ are the minimum and the maximum frequencies and c is the speed of light. Thus the conclusion from the equation 3.9 is that the number of samples required in α will clearly be proportional to the subaperture length *l*. From the equation 3.10, the sampling rate in *r* depends only on the system bandwidth, thus not the subaperture size [5].

3.2.2 Implementation

In the preceding paragraph, we focused on the mathematics and the equation derivation of the Fast backprojection. In this paragraph, we explain in detail how the Fast backprojection works, and illustrate the implementation of this algorithm. The Fast backprojection works on a pulse-by-pulse basis. In other words, we take a pulse, read it and compute it, and finally move to the next pulse [5].

The grid for the desired output image is defined. Thus, the three dimensional coordinates of every pixel must be known [5]. The pixels are arranged in a regular Cartesian grid, it is sufficient to specify the location of the corner pixel and the displacement vectors in the range and angular directions [5]. A subaperture size of $\sqrt{N}$ since it is the optimal size for an $M\times N$ image where *N* is the azimuth size. The smaller the subaperture size is, the smaller the amount of trial and error testing will be. A subaperture by subaperture basis is followed for processing [5]. So in each subaperture, we have to compute the pixel locations for a polar grid that is in the middle of the

current subaperture. The pixel spacing in the radial and angular directions is chosen to be above the Nyquist rates given in equations 3.9 and 3.10. The pulses for every subaperture will be processed by applying the backprojection technique [5]. After that we must transfer the obtained results from the polar grid to the final image Cartesian grid. This process contains several steps, starting by upsampling the polar image data in both directions, and finally by computing a contribution for every pixel in the output image by finding the corresponding point in our upsampled polar image data [5].

The interpolation is complicated due to the two dimensions to be handled when forming the final image. Consequently, the interpolation in Fast backprojection is always a 2D interpolation in the range and azimuth direction. This interpolation is therefore used to approximate the polar data in between pixels on the upsampled polar grid. If the upsampling factor is large, the results will be more accurate [5].

After performing all the steps mentioned above, we continue to the next subaperture and compute a new polar grid. This process will carry on until all the pulses have been used. The Fast backprojection has a notable advantage as large images can be created with little memory and insignificant consequences on the speed. This fact is proven since the output image data is updated only once per subaperture, thus it is reasonable to store the output image on disk rather than in memory [5]. However, the polar grid images must be stored in memory, as they require much less space even after we upsample them, being thin in the angular direction.

3.2.3 Computational Load

This section explains how the computational load equation was derived. In this thesis, and due to SAR mode used, we want to create an $M \times N$ pixel image in with L_{p} number of pulses of range compressed data. These tasks mentioned below will be repeated for each subaperture, with the corresponding operations count in parentheses [5]: we compute coordinates of polar grid, we do the backprojection, we upsample and then we interpolate polar grid to Cartesian output grid. Now the total operations count for the full image will be shown in the equation 3.11 [5] below:

$$N_{\mathrm{op}} \approx L_{\mathrm{p}} N \left(1 + m + \frac{M}{m} \right) \tag{3.11}$$

where m is selected to be equal to $\sqrt{N}$ to reduce the operations count. Two effects should be taken into consideration; if the subaperture is small, then the updated full high resolution image

will become frequent, and if the subaperture is large then the cross range resolution of the subaperture image become higher [5]. Thus, the best subaperture size has to represent conciliation between these two effects. If $M = N$, equation 3.11 will be reduced, as is shown in the equation 3.12 [5] below:

$$N_{op} = N^2 \sqrt{N} \tag{3.12}$$

3.3 FBP Implementation with Matlab

Before implementing the Fast backprojection in Matlab, we assume the SAR system parameters. We create the received data matrix for one point fixed target. To do so, we must firstly define the frequency, time, target and the platform parameters. Secondly, we have to define the transmitted signal and the matched filter parameters in order to get the received signal matrix. Table 3.1 shows the parameters used. For the frequency parameters, we assume the minimum frequency to be 20 MHz and the maximum frequency to be 90 MHz, thus the centre frequency is equal to 55 MHz and the bandwidth to 70 MHz, and set the sampling frequency to be $f_s = 4 \times f_{max}$ since the sampling frequency is always twice larger than the maximum frequency $f_s \geq 2 \times f_{max}$. For the time parameters, we assume the pulse duration to be $5\mu s$ and the Pulse Repetition Frequency (PRF) to 100, thus the chirp rate k is equal to bandwidth divided by the pulse duration $k = B/t_p$, and the sampling time is proportional to the sampling frequency.

Table 3. 1: Set parameters.

Minimun frequency f_{min}	20 MHz
Maximun frequency f_{max}	90 MHz
Center frequency f_c	55 MHz
Bandwidth B	70 MHz
Sampling frequency f_s	360 MHz
Sampling time t_s	0.00277 μs
Pulse duration t_p	5 μs

Pulse Repetition Frequency	100
Chirp rate k	14×10^{12}
Velocity of Platform Vpl	128 ms
Altitude h	1000 m
Angle α	45˚

Assuming there is a point target in the middle of the flying trajectory and we define its Cartesians coordinates (x, y, z) to be equal to the half of the flight length, to the altitude and to zero respectively since the point is on the ground.

We set the platform parameters. Following the CARABAS-II parameters, we assume the velocity of the platform to be 128 m/s and we set the altitude to one kilometre and the integration angle α to 45 degrees. The calculation of the flight trajectory length L is just a simple geometry calculation as we can see in the figure 3.1 below:

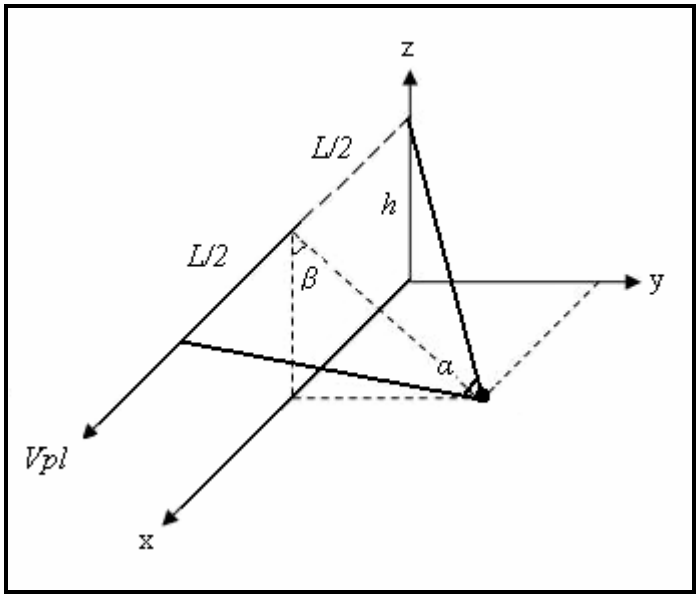

Figure 3. 1: SAR system geometry.

Since we set the Cartesian coordinate y of the target point to be equal to the altitude of the plane, the length L of the flight is calculated by the equation 3.13 shown below:

$$L = 2\sqrt{2}h \cdot \tan\left(\frac{\alpha}{2}\right) \tag{3.13}$$

The length between each sample Δ_L is calculated by dividing the velocity of the platform by the pulse repetition frequency PRF as the number of azimuth positions is calculated by:

$$N_p = \frac{L}{\Delta_L} \tag{3.14}$$

We fix the Cartesian coordinates of the plane (x, y, z) to be the vector of N_p number of positions that are separated by Δ_L, zero since the plane flies along the x-axis, and the altitude h set to one kilometre respectively.

After defining all the parameters that we need, we start calculating the range between the point target that is located in the center of the Cartesian image, and the every plane position

$$R = \sqrt{(x_{tg} - x_{pla})^2 + (y_{tg} - y_{pla})^2 + (z_{tg} - z_{pla})^2} \tag{3.15}$$

To get the received signal matrix, we have to create the transmitted signal using the formula and continually create a matched filter that is the inverse conjugate of the transmitted signal:

$$S_{tx}(\tau) = A(\tau) \cdot e^{j2\pi\left(f_c\tau + \frac{k\tau^2}{2}\right)} \tag{3.16}$$

where τ is a vector from $[-t_p/2\ \ t_p/2]$, f_c is the centre frequency, k is the chirp rate and $A(\tau)$ is the amplitude given by the radar equation. A is always equal to one, since it is the same signal transmitted. The matched filter, which is the inverse conjugate of the transmitted signal S, followed by:

$$S_{matched}(\tau) = e^{j2\pi\left(f_c\cdot(-\tau) + \frac{k\cdot(-\tau)^2}{2}\right)} \tag{3.17}$$

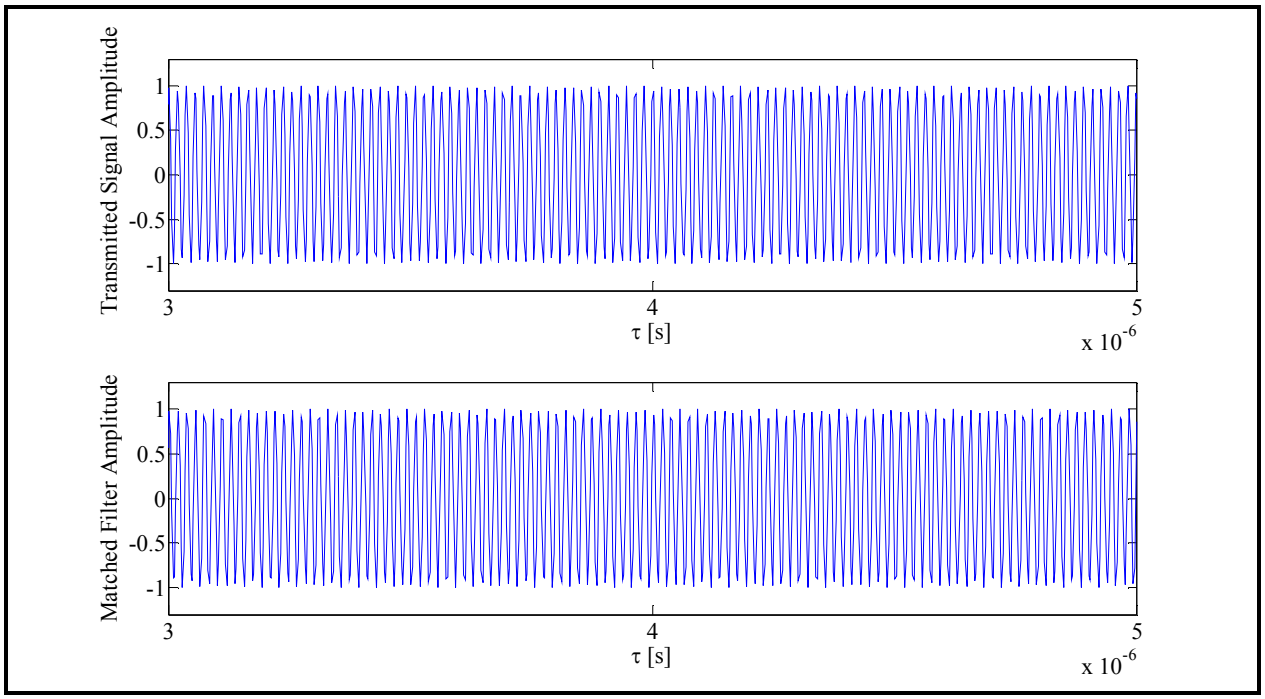

Figure 3. 2: Plot of the time domain transmitted signal and the matched filter.

Figure 3.2 shows the plot of the transmitted signal and the matched filter respectively. The received signal equation differs substantially from the transmitted signal equation, given that we will receive a new different signal for every transmitted signal. Thus, τ and the amplitude $A(\tau)$ will differ from the transmitted signal equation, since the range and time delay will be different for every received signal. Each received signal will be rectified, using a rectangular function before it is been convolved with the match filter. This process is made for every received signal and the result is saved in a matrix, called 'received data matrix'. We use this matrix to implement the Fast backprojection for SAR image retrieval. The equation of the received signal for every plane position *n* is given by equation 3.18 below:

$$S_{\mathrm{rx}}(n) = A(n)\cdot \mathrm{rect}\left(\tau - \frac{2R(n)}{\mathrm{c}}\right)\cdot \mathrm{e}^{\mathrm{j}2\pi\left(f_{\mathrm{c}}\left(\tau-\frac{2R(n)}{\mathrm{c}}\right)+\frac{k\left(\tau-\frac{2R(n)}{\mathrm{c}}\right)^2}{2}\right)} \tag{3.18}$$

where τ is vector from $[\min(2R)/\mathrm{c} - t_{\mathrm{p}}, \max(2R)/\mathrm{c} + t_{\mathrm{p}}]$, f_{c} is the centre frequency, k is the chirp rate, R is the range that is changed for every received signal and the amplitude $A(n)$ is no longer equal to one, since every received signal is different from the other. Thus, the amplitude $A(n)$ now follows the radar equation and it is equal to:

$$A(n) = \frac{1}{(4\pi)^2 R(n)^4}$$

where the range $R(n)$ changes for every received signal. $S_{rx}(n)$ is convolved with the matched filter and the result will be saved in the matrix. After we repeat this process for the entire received signal we plot the results in Matlab and find the figures 3.3 and 3.3 as indicated below.

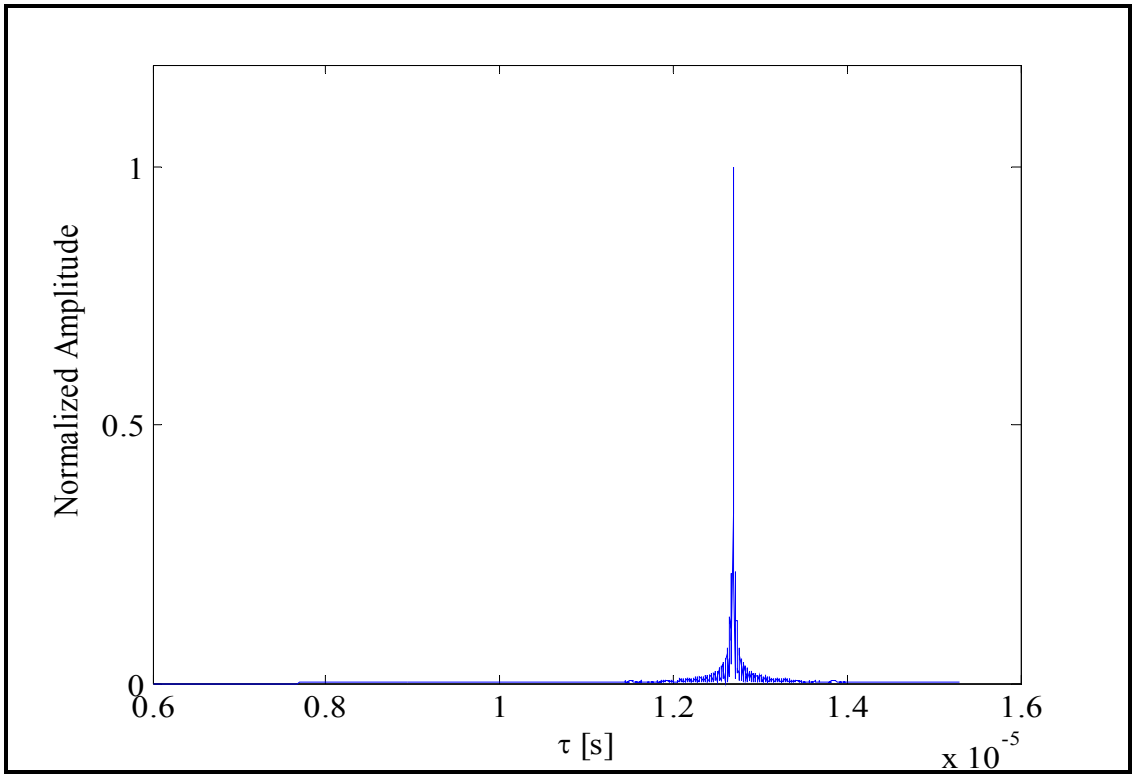

Figure 3. 3: Plot of the first received signal.

The received data matrix is the data of the point target of which we will retrieve its image. It is then now the signal processing time. The following paragraph evidences how we backproject this matrix using the Fast backprojection to retrieve the SAR image.

We will describe every step in detail, and exemplify how to divide the aperture into subapertures, how to calculate the Cartesian coordinate x of the centre of each subaperture, and how to create the polar image for every subaperture by calculating the $r_{\min}, r_{\max}, \alpha_{\min}, \alpha_{\max}$ for every polar grid. We then explain how to transfer the data from every polar image to the final Cartesian image, and how to implement different interpolation types. In conclusion, we present a small study based on the range method to compare between interpolation techniques in term of image quality.

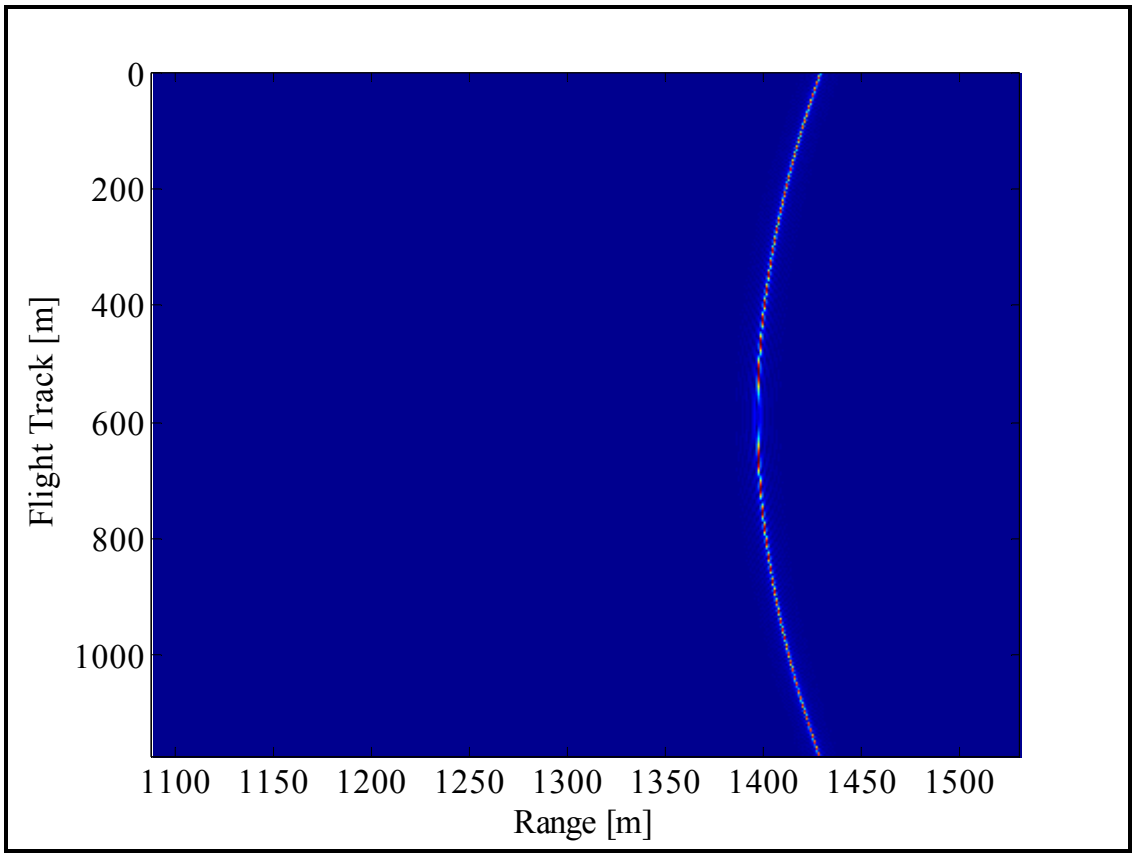

Figure 3. 4: Plot of the received signal matrix.

We create an empty image of size $N \times N$, where we set N be to equal to 256. We choose the subaperture size to be equal to $\sqrt{N}$ = 16. The aperture is then divided by the subaperture size to get the number of subapertures. Secondly, we define Δ_r and Δ_a to values that satisfy the conditions in equation 3.9 and 3.10. A ‘‘for loop’’ is then run for every subaperture where we calculate its centre coordinates, create its polar grid, retrieve its image in the polar grid and then transfer the data to the final Cartesian image. These steps will be repeated for every subaperture. We now analyse the process for one subaperture describing the steps in detail.

The subaperture is considered, by which we calculate the Cartesian coordinates of its centre position. The Cartesian coordinate x is calculated by using the ‘mean value’ method which consists of adding all the Cartesian coordinate $x(k)$ of each position in the subaperture and then dividing the sum by the number of positions following the formula 3.19 below:

$$x_{\text{center}} = \frac{x(k) + x(k+1) ..... + x(k+N_{\text{pos}})}{N_{\text{pos}}} \tag{3.19}$$

where x_{center} is the Cartesian coordinate x of the subaperture center, N_{pos} is the number of the plane positions in one subaperture, $x(k)$ is the Cartesian coordinate x of the first position of the subaperture and $x\left(k+N_{\text{pos}}\right)$ is the Cartesian coordinate x of the last position. The center point A(x, y, z) is equal to A(x_{center}, 0, h) since we consider that the flight trajectory is linear and the target is in the middle of the flight track. We suppose that the plane does not move left or right, up or down, the Cartesian coordinate x_{center} will change from one subaperture to another.

The polar grid for the first subaperture is defined. We start by defining the final image grid where we consider it in the FBP case as a slant image and the point target is in the middle of the image. The Cartesian coordinates x of the image belong to the vector $\left[L/2-N/2,\ L/2+N/2\right]$ and the polar coordinates r belong to the vector $[h\sqrt{2}-N/2, h\sqrt{2}+N/2]$. We calculate the range and the angle between the x_{center} and every pixel of the image following the equation 3.20 and 3.21, where the minimum range is set to be $r_{\min}$, the maximum range is set to be the $r_{\max}$, the minimum angle is set to be $\alpha_{\min}$ and the maximum angle is set to be $\alpha_{\max}$.

$$r_{\text{polar}} = \sqrt{\left[x_{\text{image}}(n) - x_{\text{center}}\right]^2 + r_{\text{image}}(n)^2} \tag{3.20}$$

$$\alpha_{\text{polar}} = \frac{x_{\text{image}}(n) - x_{\text{center}}}{r_{\text{polar}}} \tag{3.21}$$

From these two equations we can define the polar image of every subaperture. The first step is to retrieve the image of each subaperture in the polar grid and the second step is to transfer these data to the final image grid. We will not use any interpolation techniques; we simply use a round function to approximate the polar image data. In the following paragraphs, we will explain every interpolation technique in detail. The task is to describe how the polar image is created, and how to transfer the data to the final image. To create the polar image of the subaperture, we must firstly create an empty image of size $N_{\text{r}} \times N_{\alpha}$ where N_{r} is the number of range positions and N_{α} is the number of the angular positions in the polar grid, where they are calculated by:

$$N_{\text{r}} = \frac{r_{\max} - r_{\min}}{\Delta_{\text{r}}} \tag{3.22}$$

$$N_{\alpha} = \frac{\alpha_{\max} - \alpha_{\min}}{\Delta_{\alpha}} \tag{3.23}$$

N_{α} is always a small number due to the polar grid being thin in the angular direction . We now run another 'for loop' to calculate the distance between every position of the plane in each subaperture, with every pixel position in the polar grid. The Cartesians coordinates of one point in the polar grid are shown:

$$\begin{aligned} x_{\text{po}} &= r \cdot \cos\alpha \\ y_{\text{po}} &= r' \cdot \sin\beta \\ z_{\text{po}} &= r' \cdot \cos\beta \end{aligned} \tag{3.24}$$

where *r* is the range and α is the angle, $r' = r \cdot \sin\alpha$ and $\beta = \tan^{-1}\left(\frac{y_{\text{tg}}}{h}\right)$. The distance between every plane position and the every pixel position in the polar grid will be calculated by:

$$D = \sqrt{[(x_{\text{pla}}(u) - x_{\text{center}}) - x_{\text{po}}]^2 - [(y_{\text{pla}} - y_{\text{center}}) - y_{\text{po}}]^2 - [(z_{\text{pla}} - z_{\text{center}}) - z_{\text{po}}]^2} \tag{3.25}$$

The Cartesian coordinates of the plane for every plane position are $\mathrm{P} = \left(x_{\text{pla}}, y_{\text{pla}}, z_{\text{pla}}\right)$ where x_{pla}, y_{pla} and z_{pla} are equal to the Cartesian coordinates *x* of every position on the flight track, zero and *h*, respectively. From this distance we can easily calculate the time delay by simply using the formula $t_{\text{delay}} = 2D/\mathrm{c}$. We can calculate the number of delays by dividing the delay time by the sampling time t_{s}, and then round the answer instead of using any interpolation technique. This method is the simplest way to approximate the received data *S*. After rounding the division, we search the value of n_{delay} in the array of this specific position from the received data matrix *S* and we add the value in the SAR polar image matrix. We repeat this step for the sixteen positions of the plane in each subaperture. The plot in figure 3.5 shows the image of the first subaperture in the polar grid.

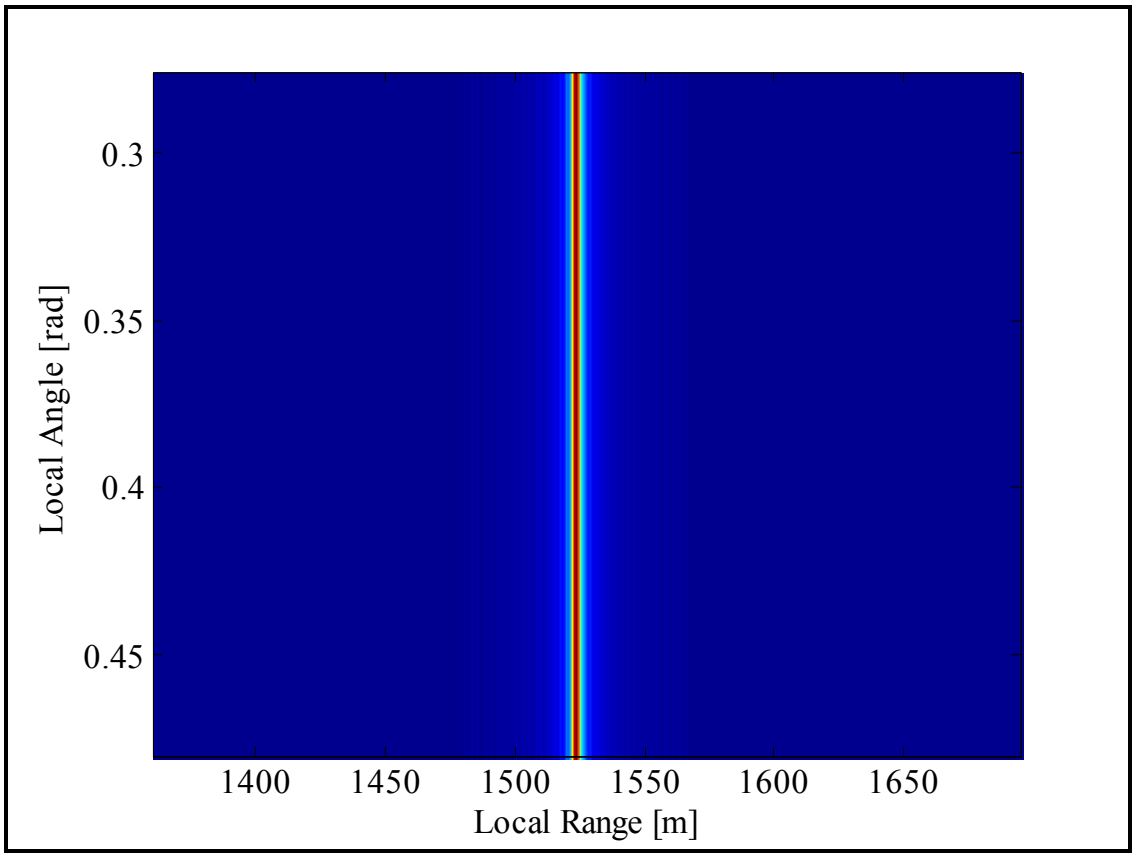

Figure 3. 5: SAR polar image for the first subaperture.

The polar data on the polar grid is transferred to the final image grid that we have previously defined. To do so, we must firstly calculate the range *r* between the x_{center} of each subaperture and every pixel in the final image grid. Then, from every range *r*, we can calculate the angle value *α*. The following equations showed how the range and the angle are calculated.

$$r_{\text{term}} = \sqrt{\left[x_{\text{center}} - \left(\frac{L}{2} - \frac{N}{2} + i\right)\right]^2 + \left[y_{\text{center}} - \left(h\sqrt{2} - \frac{N}{2} + j\right)\right]^2} \tag{3.26}$$

$$r = \frac{r_{\text{term}} - r_{\min}}{\Delta_{\text{r}}} \tag{3.27}$$

$$\alpha_{\text{term}} = \frac{\left(x_{\text{tg}} - \frac{N}{2} + i\right) - x_{\text{center}}}{r_{\text{term}}} \tag{3.28}$$

$$\alpha = \frac{\alpha_{\text{term}} - \alpha_{\min}}{\Delta_{\alpha}} \tag{3.29}$$

For every *r* and α, we search their values in the SAR polar image matrix, we retrieve them and put them in the SAR final image matrix. Ultimately, we repeat these steps for every subaperture and we will get the final image matrix that is plotted with its transfer function in the frequency domain. These plots are highlighted in the figures 3.6 and 3.7 below.

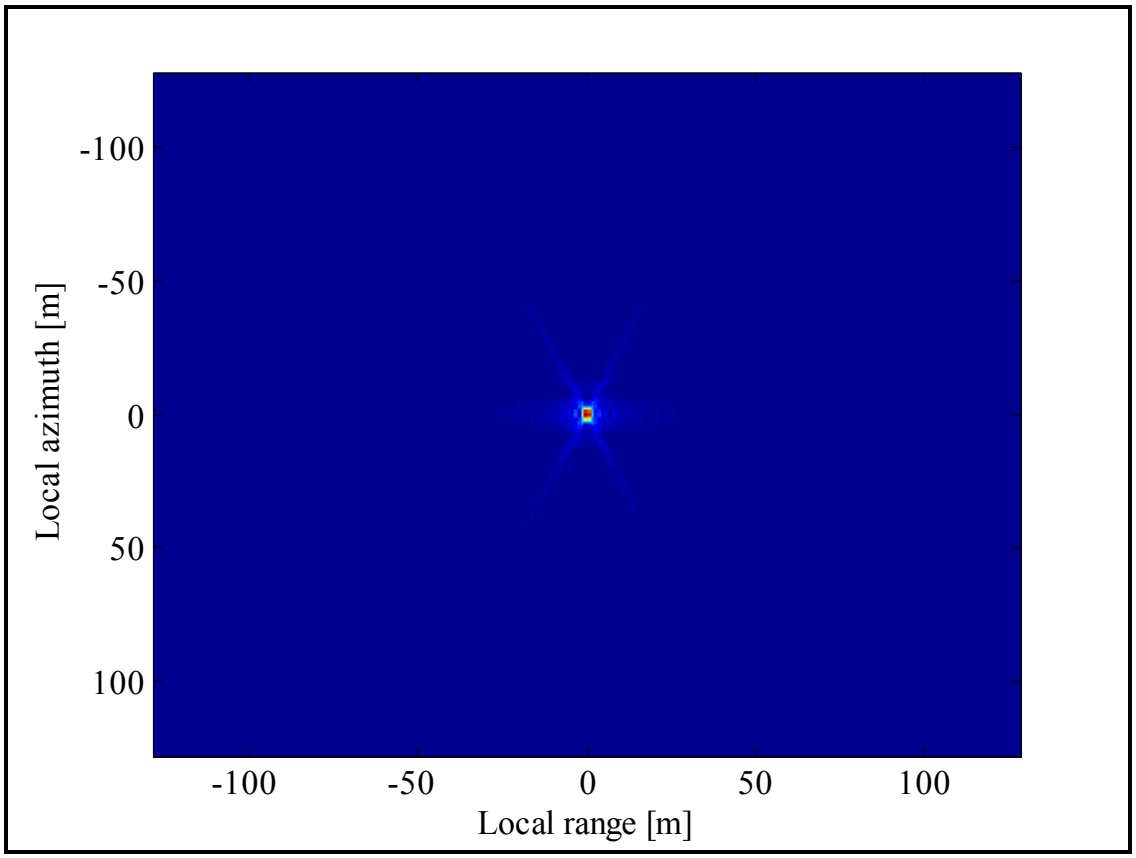

Figure 3. 6: FBP image of one point target without interpolation.

Examining the image, we can conclude that it is not a high-quality image and there is a lot of noise. This is due to the use of the round function as a substitute for interpolation techniques to approximate the polar image data between the pixels. The figure 3.7 is even worse in quality and contains a lot of noise. To resolve this issue, we will explain in the following paragraph the three interpolation techniques that we used to retrieve better quality image.

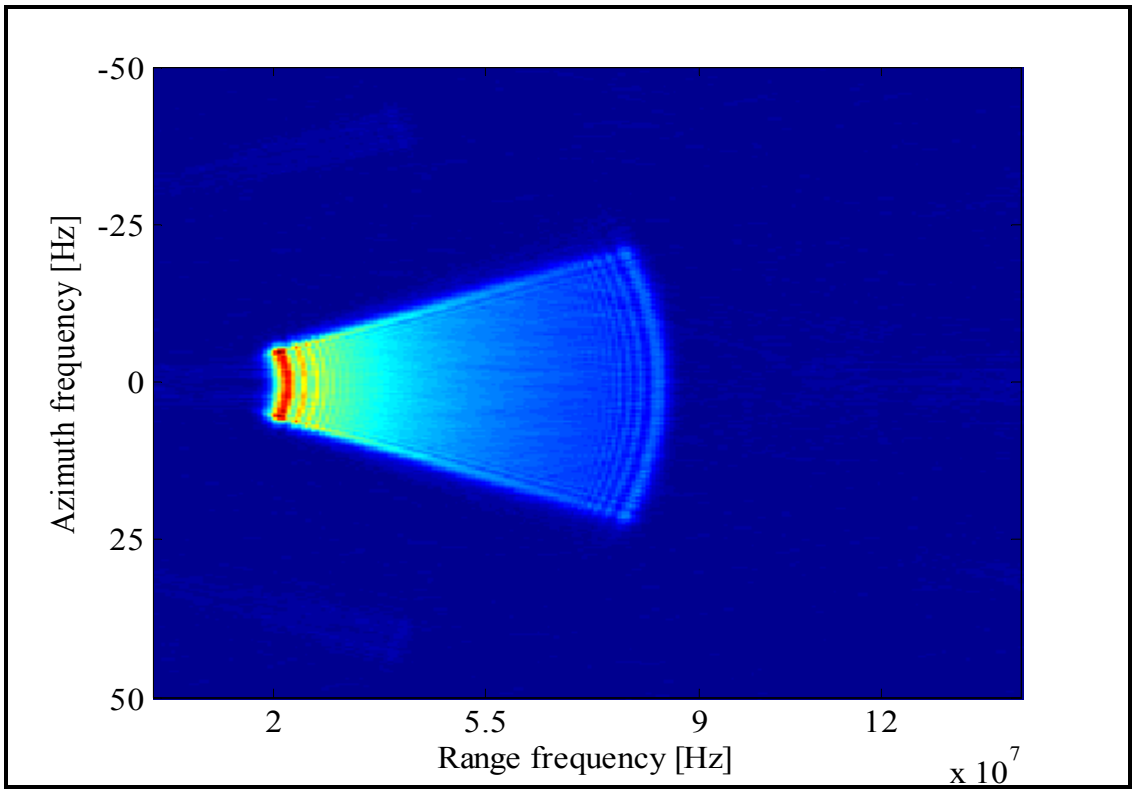

Figure 3. 7: System transfer function without interpolation.

3.3.1 Implementation of Double Linear Interpolation

Since the quality of the final image was insufficient, and the transfer function image contains substantial noise, we had to again implement the Fast backprojection with a double linear interpolation to approximate the polar image data. A new empty image of size $N \times N$ is now created, where N is equal to 256. We repeat the same step as explained before to calculate the coordinates of every subaperture centre, the $r_{\min}, r_{\max}, \alpha_{\min}, \alpha_{\max}$, thus defining the polar grid for every subaperture. To create the polar image matrix, we used a one-direction linear interpolation to approximate the received data in the matrix S. We upsample the matrix S in the range direction by an upsampling factor and calculate the distance and the time delays using the same equation 3.15. However, when now calculating the number of delays, we have not used the function round, instead implementing the linear interpolation equation 2.5, explained previously in Chapter 2. To do so, we first multiply the fraction between the time delay and the sampling time by the upsampling factor, then use the function ‘‘floor’’ to floor the number of delays. η or the desired delay value will be the difference between the numbers of delays and the floored number of delays. After ascertaining the value of η, we apply:

$$h_{\mathrm{p}} = (1-\eta)\cdot S(n) + \eta \cdot S(n+1) \tag{3.30}$$

After obtaining the interpolated SAR polar image matrix, we upsample it in the range direction and the angular direction using the same upsampling factor. We then follow the same step and equations as used before to calculate r and α. Here, instead of using the round function for approximation, we will interpolate in two directions to get a better approximation, therefore a better quality image. So first η_1 and η_2 is calculated in the same way as calculating η and then we use the 2D linear interpolation equation 3.31 to create the final image matrix.

Interpolation in range

$$h_{\mathrm{r}}(n,m) = \eta_1 \cdot \left[h_{\mathrm{p}}(n+1,m) - h_{\mathrm{p}}(n,m)\right]$$

Interpolation in azimuth

$$h_{\mathrm{x}}(n,m) = \eta_2 \cdot \left[h_{\mathrm{p}}(n,m+1) - h_{\mathrm{p}}(n,m)\right]$$

Interpolation in azimuth and range

$$h_{\mathrm{r,x}}(n,m) = \eta_1\eta_2 \cdot \left[h_{\mathrm{p}}(n+1,m+1) + h_{\mathrm{p}}(n,m) - h_{\mathrm{p}}(n,m+1) - h_{\mathrm{p}}(n+1,m)\right]$$

2D linear interpolation

$$h(n,m) = h_{\mathrm{p}}(n,m) + h_r(n,m) + h_x(n,m) + h_{r,x}(n,m) \tag{3.31}$$

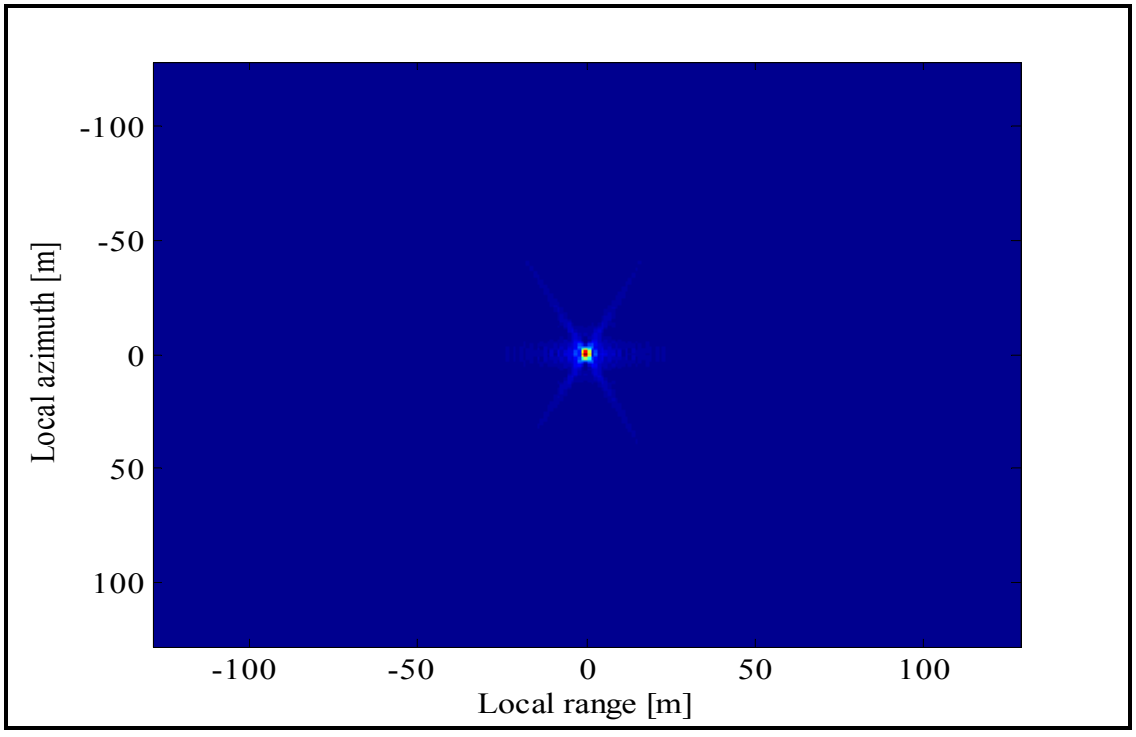

Figure 3. 8: FBP image of one point target with double linear interpolation.

where $h(n,m)$ is the final image data, $h_p(n,m)$ is the polar data matrix and n, m are the floored values of α and r respectively. After repeating the same steps for the entire subapertures, the final image matrix is then plotted with its transfer function in the frequency domain as it is shown in the figures 3.8 and 3.9 respectively.

From the figure below, we can readily confirm that the image quality, when implementing linear interpolation in the range and angular directions been strongly improved. One can say that the noise in the previous images has almost vanished. This conclusion is made easily, as we have compared two images, one implemented without interpolation and the second with interpolation. However, later in this chapter, we will describe two other variants of interpolation, which therefore makes the comparison between the final images difficult to finalise by simply viewing the images. The range resolution calculation method will hence be further addressed to compare interpolations and conclude which is the most effective interpolation to use achieve accurate approximation and a better quality image.

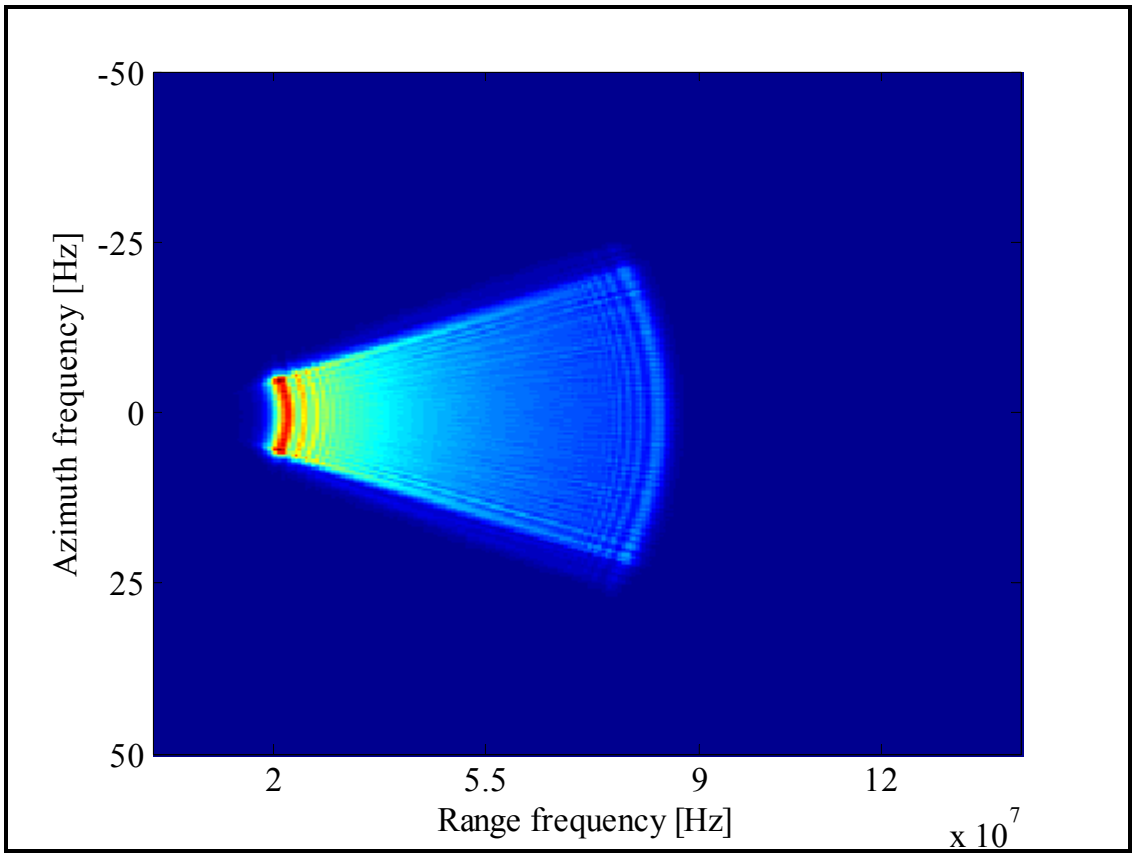

Figure 3. 9: System transfer function with double linear interpolation.

3.3.2 Implementation of Double Triangular Interpolation

The linear interpolation is a satisfactory method to use since it generates a high quality image in an acceptable processing time, but the challenge in this thesis was to implement different types of interpolation that give more accurate results, and therefore optimal image. For this purpose, we now introduce the double triangular interpolation. This method consists of convolving the data with a triangular function. This interpolation is more complicated and difficult to implement, because it is necessary to interpolate in the range and angular direction in the same timeframe. The process will now be explained in steps as to how it is implemented in Matlab. Firstly, we create a new empty image of size $N \times N$ where N is equal to 256, and we calculate the coordinates of every subaperture centre following the same formula used before, and define the polar grid by calculating the values of $r_{\min}, r_{\max}, \alpha_{\min}, \alpha_{\max}$. Secondly, we retrieve the polar image matrix by upsampling the received data matrix, and then interpolating it using a one-directional interpolation as explained in the preceding section. After we have obtained the interpolated SAR polar image matrix, we upsample it in the range direction and the angular direction using the same upsampling factor. We calculate the range, *r*, and the angle, α, in the same way, using the previously applied formulas. From *r* and α, we can easily calculate η_1 and η_2 where we will convolve them with a triangular function explained in Chapter 2. Following this, the formula 2.7 is now changed from *time delay* to *number of delays*, following the formula 3.32 below:

$$p_s(n) = \begin{cases} 1 - |\eta| & \text{for } |\eta| \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.32}$$

which will lead to the formula 3.33, showing that it is necessary to convolve the received data matrix with the positive part of the triangle, defined in Matlab with the array $p_s = 1 - [0 : \Delta : 1]$ where Δ is equal to 0.01.

$$p_s(n) = \begin{cases} 2 - \eta, & \text{for } |\eta| \le 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.33}$$

This convolution was difficult to implement it in Matlab, being done in two directions. Finally, after numerous trials, we generated the formula, 3.34, below that gives a very accurate result and a good quality image.

Interpolation in range

$$h_r(n,m) = p_s\left(\frac{\eta_1}{\Delta}\right) * \left[h_p(n+1,m) - h_p(n,m)\right]$$

Interpolation in azimuth

$$h_{\mathrm{x}}(n,m) = p_{\mathrm{s}}\left(\frac{\eta_2}{\Delta}\right) * \left[h_{\mathrm{p}}(n,m+1) - h_{\mathrm{p}}(n,m)\right]$$

Interpolation in azimuth and range

$$h_{\mathrm{r,x}}(n,m) = p_{\mathrm{s}}\left(\frac{\eta_1}{\Delta}\right) \cdot p_{\mathrm{s}}\left(\frac{\eta_2}{\Delta}\right) * \left[h_{\mathrm{p}}(n+1,m+1) + h_{\mathrm{p}}(n,m) - h_{\mathrm{p}}(n,m+1) - h_{\mathrm{p}}(n+1,m)\right]$$

2D triangular interpolation

$$h(n,m) = h_{\mathrm{p}}(n,m) + h_r(n,m) + h_x(n,m) + h_{r,x}(n,m) \tag{3.34}$$

where $h(n,m)$ is the final image data matrix, $h_p(n,m)$ is the polar data matrix, n and m are the floored values of α and r respectively, and p_{s} is the triangular function. As can be seen from the formula, to interpolate, we had to convolve the triangular function with two nearest values from r and α, where the first value is separated from the real value of r by η_1 and the second by 1- η_1. The same principle applies for the value of α, where we had convolved with the value of η_2, 1-η_2. The figures 3.10 and 3.11 below demonstrate the plot of the point target and its transfer function.

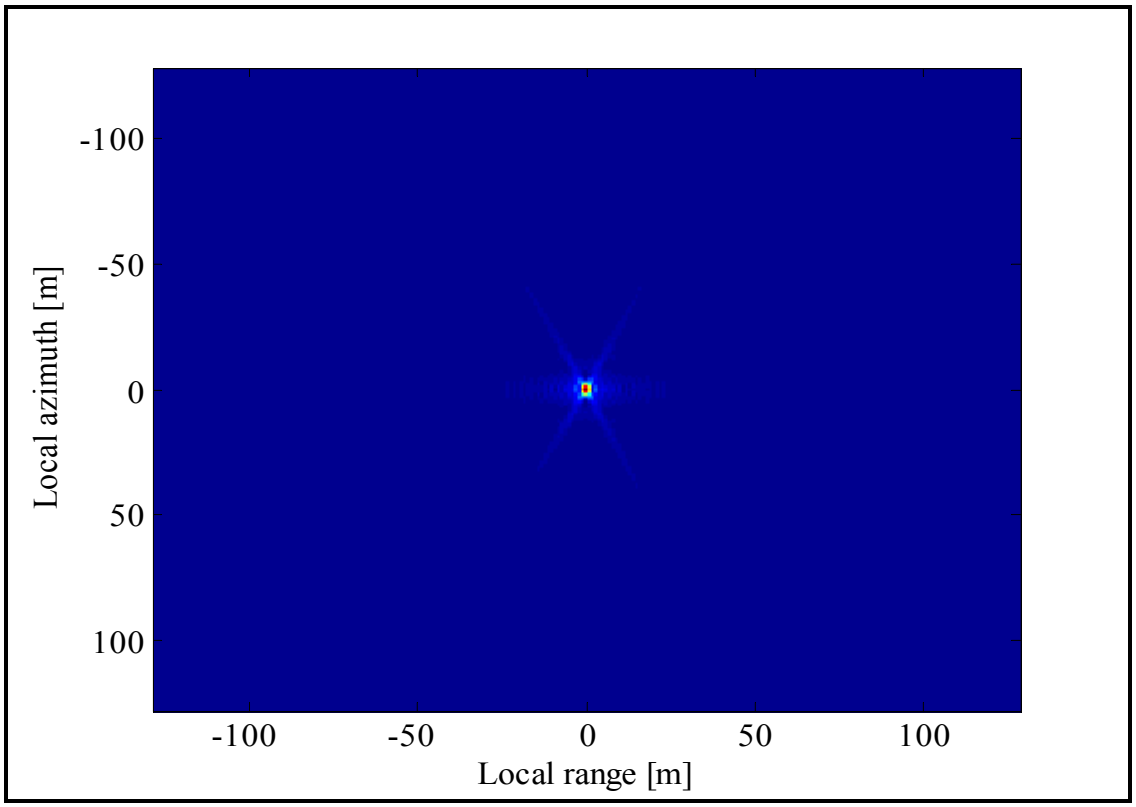

Figure 3. 10: FBP image of one point target with double triangular interpolation.

The images are clear and do not contain much noise; consequently, one can say that this interpolation technique is sufficient to implement. But to definitively determine if this technique was better than the linear, one we had to perform the range method. The result will be given at the end of this chapter after discussing the 2D sinc interpolation. This will lead to our conclusion as to which type of interpolation was the best to use regarding image quality.

As the interpolation becomes more and more complicated, the image retrieved will be clearer and the processing time will become elongated. This is the trade-off between the image quality and the processing time. The upsampling factor has a significant effect on the processing time as well as on the computational load. In the comparison chapter, we will present a small study on how different interpolations and different upsampling factors will affect the processing time. A small timetable will be presented to demonstrate the time difference between the Global backprojection and the Fast backprojection for every interpolation type explained in this thesis, and for different upsampling factors used. Finally, we will conclude which time domain algorithm gives the best quality image in the fastest processing time.

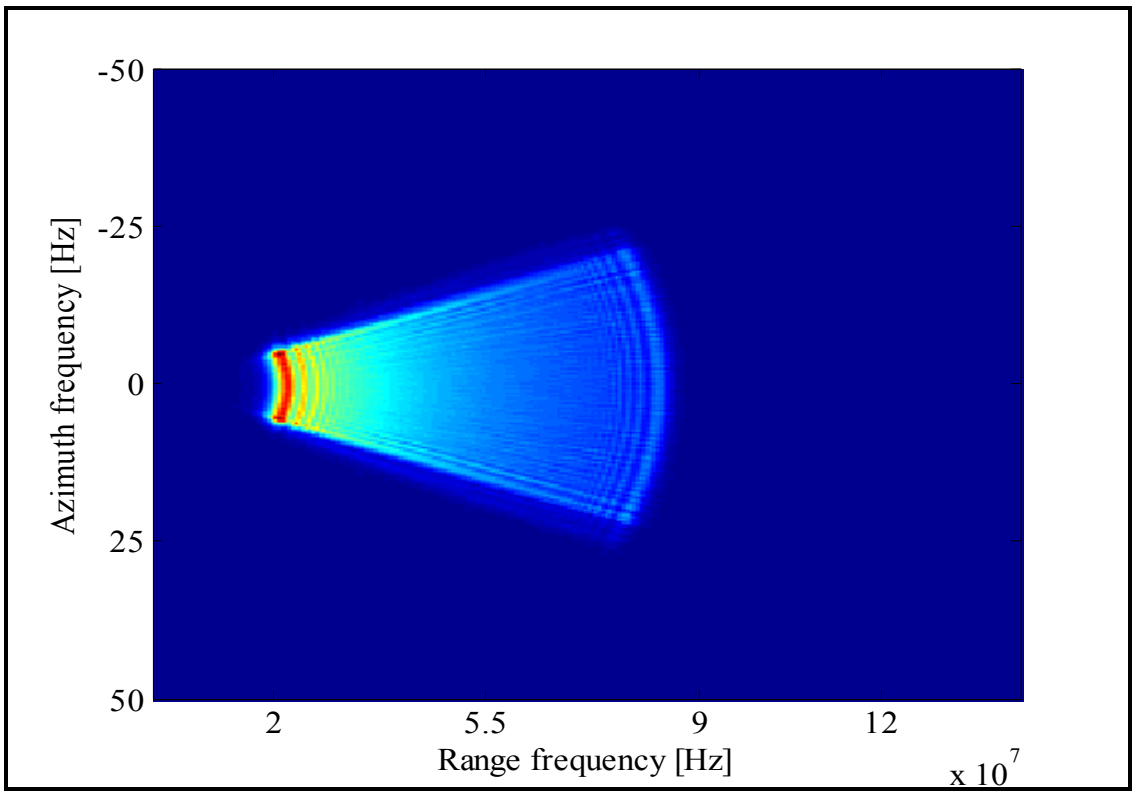

Figure 3. 11: System transfer function with double triangular interpolation.

3.3.3 Implementation of Double Sinc Interpolation

The final interpolation requiring invocation is the 2D sinc interpolation. The sinc interpolation was the hardest interpolation to actuate due to its complexity. It is a two-directional interpolation in range, and angular. It is expected to be the optimum interpolation to use in retrieving a best quality image among other techniques. The process for employing it in Matlab will now be illuminated. Firstly, we create an $N \times N$ image where N is equal to 256 then we repeat all the steps explained in the previous section, in order to calculate the SAR polar image matrix, r and α, and then η_1 and η_2 where we will now convolve them with the sinc function that we have already defined in Matlab in the array shown in the formula 3.35 below:

$$\text{sinc}(\varphi) = \frac{\sin(\pi\varphi)}{\pi\varphi} \quad \text{where } \varphi = [-2\pi : \Delta : 2\pi] \tag{3.35}$$

where Δ is the step size and it is equal to 0.01. This convolution differs from the one in the previous section, as we had to convolve the sinc function with twelve nearest values from r and α, where the first value is separated from the real value of r by η_1 and the second by 1- η_1 and so on. The same principle applies for the value of α, we will convolve the sinc function with the first twelve nearest points of this value α. The formula that we implemented in Matlab to interpolate the values of r and α to get more accurate result is found below in the formula 3.36:

Interpolation in range

$$h_{\text{r}}(n,m) = \sum_{k=0}^{11} \text{sinc}\left(\frac{k-\eta_1}{\Delta}\right) * \left[h_{\text{p}}(n+k+1, m+k) - h_{\text{p}}(n+k, m+k)\right]$$

Interpolation in azimuth

$$h_{\text{x}}(n,m) = \sum_{k=0}^{11} \text{sinc}\left(\frac{k+1-\eta_2}{\Delta}\right) * \left[h_{\text{p}}(n+k, m+k+1) - h_{\text{p}}(n+k, m+k)\right]$$

Interpolation in azimuth and range

$$h_{\text{r,x}}(n,m) =$$

$$\sum_{k=0}^{11} \text{sinc}\left(\frac{k+1-\eta_1}{\Delta}\right) \cdot \text{sinc}\left(\frac{k-\eta_2}{\Delta}\right) * \left[h_{\text{p}}(n+k+1, m+k+1) + h_{\text{p}}(n+k, m+k) - h_{\text{p}}(n+k, m+k+1) - h_{\text{p}}(n+k+1, m+k)\right]$$

2D sinc interpolation

$$h(n,m) = h_{\text{p}}(n,m) + h_r(n,m) + h_x(n,m) + h_{r,x}(n,m) \tag{3.36}$$

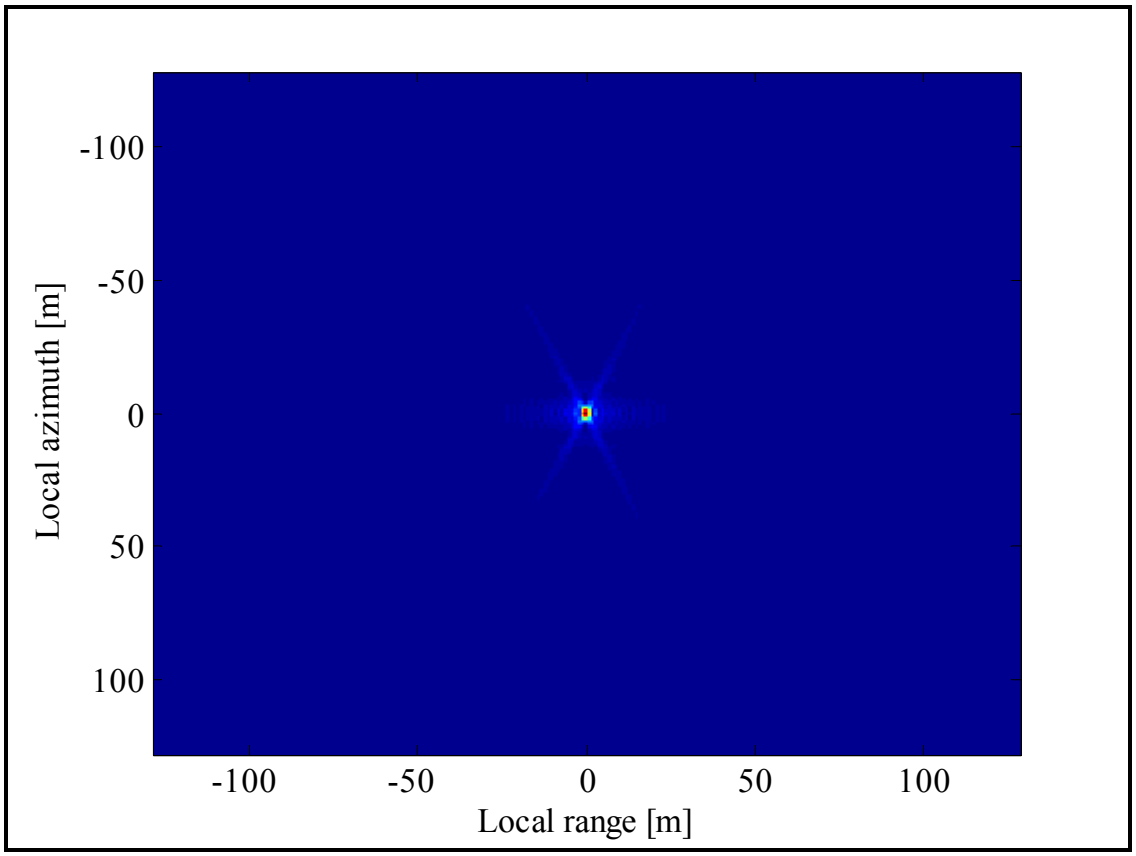

Figure 3. 12: FBP image of one point target with double sinc interpolation.

where $h(n,m)$ is the final image, $h_p(n,m)$ is the polar image matrix, $\mathrm{sinc}(\varphi)$ is the sinc function as we defined it in Matlab and n,m are the floored values of r and α.

This formula is long and needs a lot of operation to be implemented. However, the processing time to retrieve the final image was acceptable and much faster then the processing time in the Global backprojection, using the same interpolation method and the same upsampling factor. This formula is implemented and gives the image of the point target and its transfer function in frequency domain in the figures 3.12 and 3.13 below.

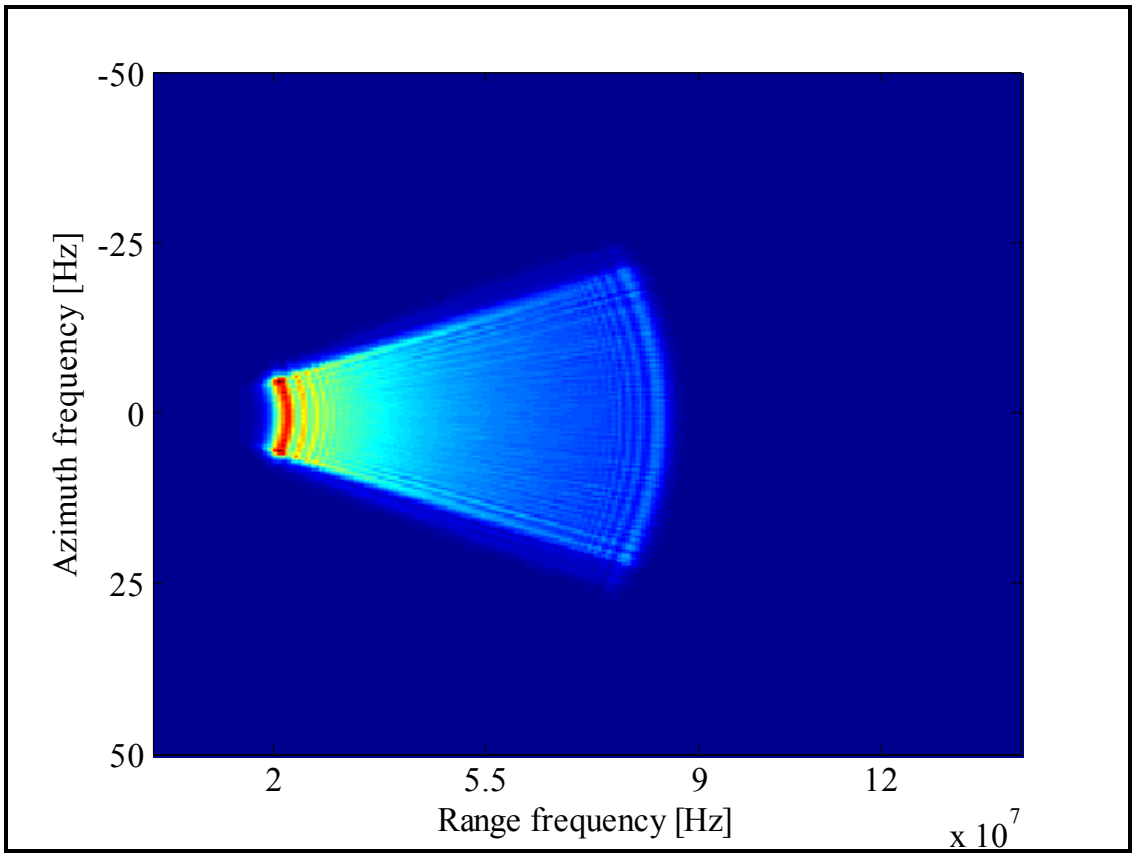

Figure 3. 13: System transfer function with double sinc interpolation.

The images are clearer then the images retrieved before. This conclusion will be presented in the following section whereby we will engage the range resolution calculation method.

3.3.4 Results and Conclusion

After implementing the three different types of interpolation, it is now appropriate to conclude which interpolation was the most effective in order to retrieve the best quality image. We will therefore employ the range resolution method comparing interpolations regarding image quality. This method consists of calculating the range of the point target from the different SAR image data which we had already obtained. Initially, we upsample the final image matrix in the range direction by a factor of 50 or 100. We then continue to plot a normalised and logarithmic SAR image data for the point target, located in the middle of the image. We measure the distance at – 3dB positions by simply using the cursor. The difference in distance is then divided by the same upsampling factor. The answer is then the exact range value. The smaller distance leads to an accurate and clear image. This method is simple, but lucidly demonstrates which interpolation is better to use in term of image quality. Table 3.2 below sets out the results ascertained after applying this method for FBP, with and without interpolation.

Table 3. 2: FBP results of the range resolution for linear track.

	-3dB left	-3dB right	Range resolution in m
No Interpolation	12537	12779	2.42 m
Linear Interpolation	12540	12780	2.40 m
Triangular Interpolation	12540	12780	2.40 m
Sinc Interpolation	12543	12781	2.38 m

From this table, we can conclude that the 2D sinc interpolation leads to the finest and clearest image, since the range distance was the shortest among other interpolation techniques. It additionally shows that when there is no interpolation, the image quality is the worst. This is an obvious conclusion, even without the use of this table as the round function is the worst function used in approximation between points.

As explained in the following comparative chapter, the sinc interpolation gives the best image, but not the fastest processing time. We will present a processing timetable that will evidence a final conclusion of which algorithm and interpolation to use in order to achieve a faster image or a clearer image.

However, before we cease this chapter, we address the motion compensation, wherein we consider that the flight track is no more linear as the case has been, in all preceding explanations. It is quite a simple issue to resolve. It is related only to the coordinates of the plane where we considered that the Cartesian coordinate y is always zero, which means that the plane does not move left or right, and the Cartesian coordinate z always is equal to the altitude h. This entails that the plane does not move upwards or downwards. Therefore, to clarify the motion compensation, we have considered that the Cartesian coordinate y move randomly in a range of [-30, 30] meters, as well as the Cartesian coordinate z moving randomly in a range of [-30, 30] metres. This slight change did not significantly affect the Matlab code. It only affects the distance or range calculation, since the Cartesian coordinates (x, y, z) change for every plane position. As we are using the time domain algorithms, this is not a complex problem to solve. This is due to using the Cartesian formula 3.15, which is not the case when dealing with the frequency domain algorithms. The conclusion made is that the motion compensation problem is fixed more easily in the time domain rather than in the frequency domain.

Again, we utilise the range method and get the same distances as for a linear trajectory. Table 3.3 below indicates the results of the range method, concluding that the motion compensation problem is properly solved.

Table 3. 3: FBP results of the range resolution for non-linear track.

	-3dB left	-3dB right	Range resolution in m
No Interpolation	12537	12779	2.42 m
Linear Interpolation	12540	12780	2.40 m
Triangular Interpolation	12540	12780	2.40 m
Sinc Interpolation	12543	12781	2.38 m

Chapter 4: A Comparison between FBP and GBP

In the previous chapter, we give a detailed explanation on how the Fast backprojection works and how to implement it with Matlab. Before that, we clarified how the received data matrix is created. In this chapter, we will use the same received data matrix and integrate it by using the GBP. We explain the GBP implementation, and the different interpolation techniques used. Finally, we will present a comparison of these time-domain algorithms in terms of image quality and processing time.

4.1 GBP Implementation in Matlab

The implementation of the backprojection is a computation in a pixel-by-pixel manner. Thus we select a pixel, we integrate over the aperture to compute its value, and then continue to the next pixel. So for each aperture position, *s*, we have a pulse $F(s,t)$ that makes a contribution toward every pixel in our desired image. In other words, we are calculating the pixels values in parallel, rather than serially. In practice, working pulse-by-pulse has a large advantage. That is, we need to access each pulse only once, and this benefits the interpolation. The filtering step can then be performed using a time domain FIR filter or simply using FFT. The integration requires performing an interpolation of $F(s,t)$ with respect to *t* using one of the interpolation methods explained in Chapter 2. Finally, we must remove the carrier phase from $F(s,t)$ prior to upsampling and interpolating, and then restore it when performing the backprojection sum. The computational load of the Global backprojection to form an $N \times N$ pixel image with *N* number of pulses of range compressed data is equal to N^3. Here, we ignore the operations required for filtering and upsampling since the Global backprojection is relatively slow, and one hardly notices the additional operation time of filtering and upsampling [5].

In implementing the Global backprojection, we will use the same received data matrix *S*, the creation of which has been explained previously in section 3.3 in Chapter 3. This received data matrix is the data of a one fixed-point target, where we will try to retrieve its image by using the Global backprojection. This point target has the same characteristics, same coordinates and same integration angle as the one used in the previous chapter. First of all, we created an empty $N \times N$ image of size 100. Then, we run three for loops; the first is the main for loop for every azimuth

position of the plane, the second is the Cartesian coordinate x of the image which is parallel to the azimuth direction, and the third is the Cartesian coordinate y of the image or the range direction. Hence, the first azimuth position of the plane together with the first Cartesian coordinates x and y pixel of the image are to be found. To do this, we calculate the distance between the position of the plane, and the first pixel of the image, using the distance formula for Cartesian coordinates in equation 3.15. After ascertaining the distance, we can calculate the time delay using the formula $t_{\text{delay}} = 2R/\text{c}$ where R is the distance or range, and c is the speed of light. Then, from the time delay calculated, we can find the number of delays by simply dividing the time delay with the sampling time *ts*. Finally, we replace the value of the first azimuth position and the value n of the number of delay in the received signal matrix, extract the value, and add it in the SAR image matrix. Since the division of the delay time by the sampling time is not an integer, we use the ‘‘round’’ function to round it up. This is the easiest route of approximation, but as we will explain later in this chapter, we will perform three different types of interpolation that will give an accurate approximation. We repeat this process for all the pixels in the image and for the entire azimuth positions of the plane. After doing this, we will get the SAR image matrix of our target point that is plotted and shown in the figure 4.1 below. Figure 4.2 indicates the system transfer function in the frequency domain after the SAR processing.

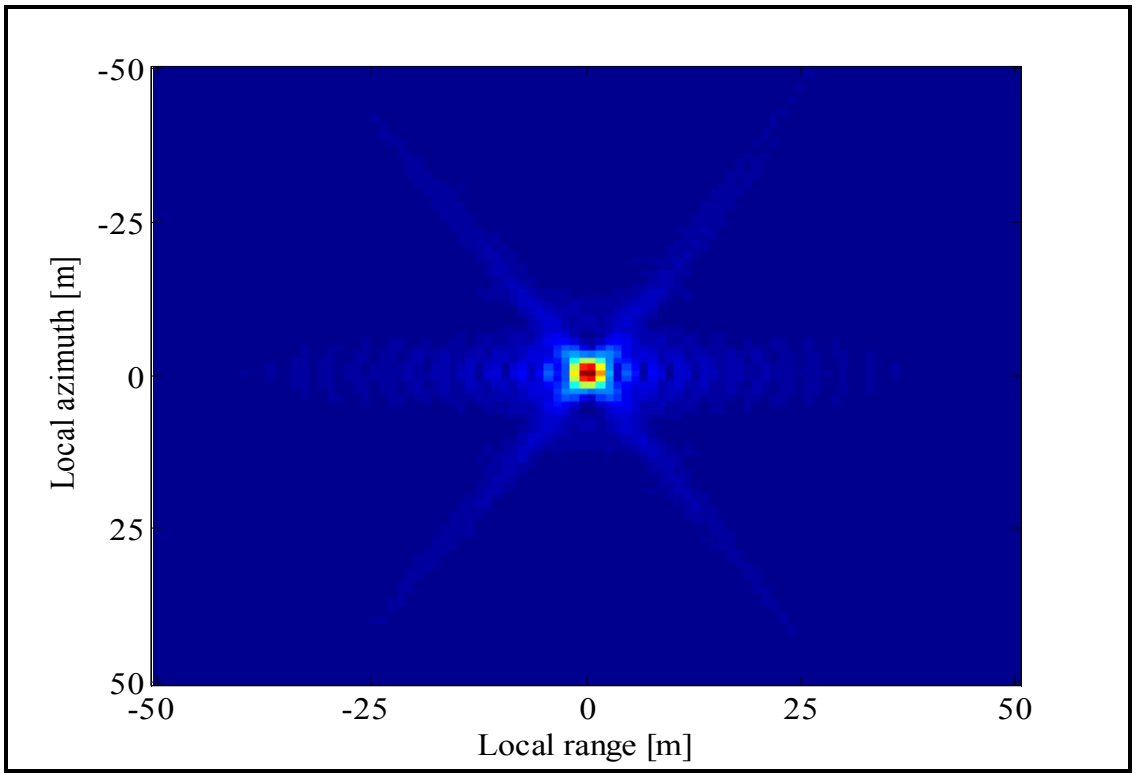

Figure 4. 1: GBP image of one point target without interpolation.

As we can see from the SAR image and the transfer function image below, these images are not very clear and contain a lot of noise. This is due to the rounding function in approximating the number of delays. As aforementioned, we will fix this problem by using three different types of interpolations: the linear interpolation, the linear interpolation with convolution and finally the sinc interpolation.

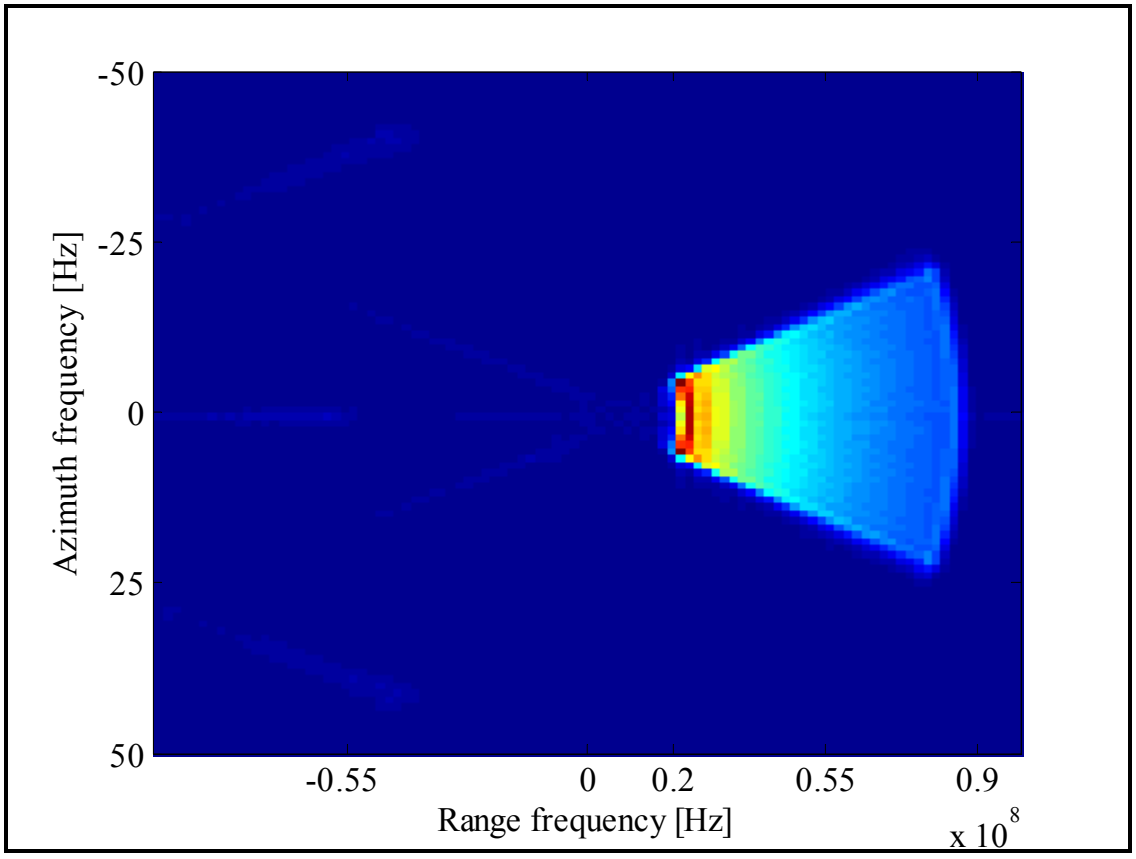

Figure 4. 2: System transfer function in frequency domain without interpolation.

4.4.1 Implementation of Linear Interpolation Technique

Since the image was not sufficiently clear, we had to implement the SAR backprojection method again, this time by using the linear interpolation concept and equation which were explained in chapter 2. Consequently, instead of using the rounding function, we will implement interpolation instead to get more accurate values that will reduce the noise in the images.

Firstly, we create another $N \times N$ empty image of size 100. Since we are invoking interpolation here, the upsampling must be used for more accurate and clear images. We therefore upsample the received signal matrix S in one direction by any chosen factor. We now repeat the same step

as explained before, to calculate the distance, and then the time delay for every azimuth position of the plane and for every pixel of the image. At this point the number of delays, which is the fraction between time delay and the sampling time, is multiplied by the upsampling factor. Here, we use the function ‘’floor’’ to floor the number of delays. η or the desired delay value is the subtraction between the numbers of delays and the floored number of delays. Finally, we apply the formula 2.5 mentioned in chapter 2, in order to get the SAR image matrix for one point target with linear interpolation. The Matlab formula 4.1 for implementing the linear interpolation is shown below:

$$h(n) = (1-\eta)\cdot S(n) + \eta \cdot S(n+1) \tag{4.1}$$

where $h(n)$ is the image matrix, S is the received data matrix, and n is the number of delays. Figures 4.3 and 4.4 respectively show the SAR image of one point target with linear interpolation and its system transfer function in the frequency domain.

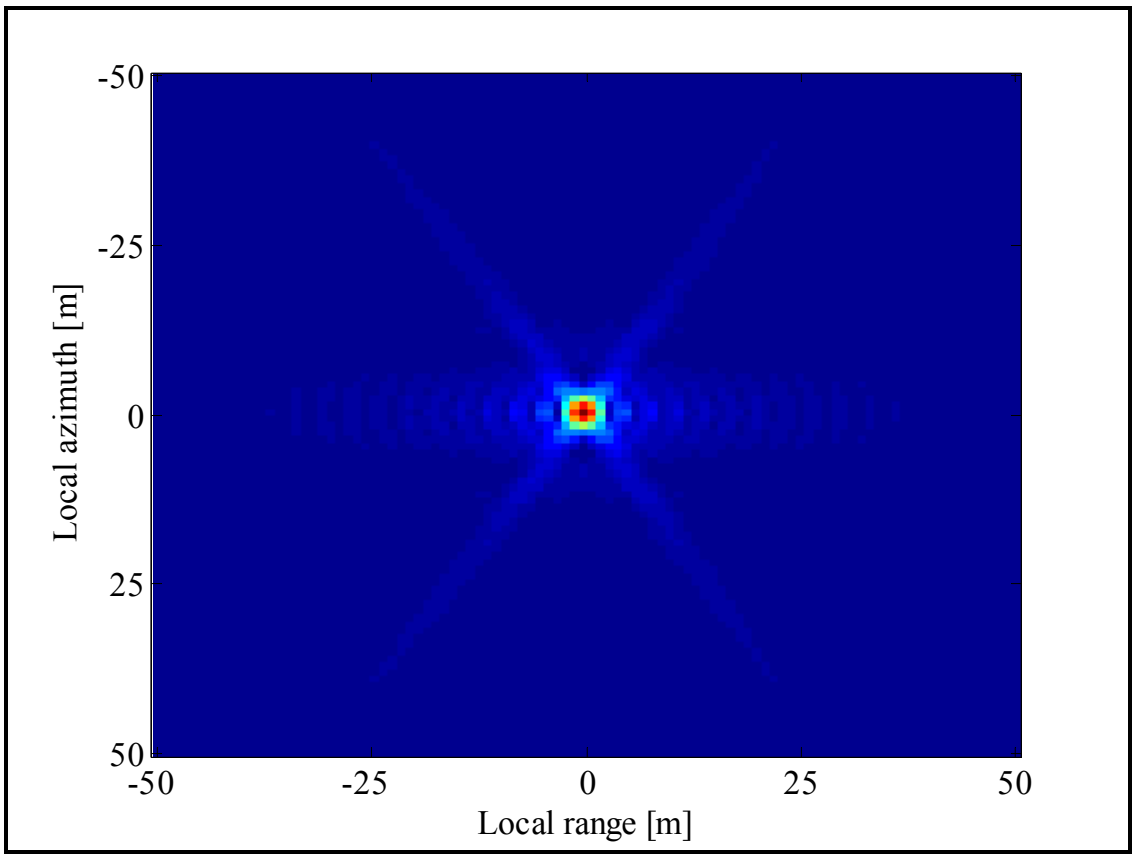

Figure 4. 3: GBP image of one point target with linear interpolation.

If we compare the two SAR images and the two system transfer functions images, we can conclude definitively that the noise is reduced, and when implementing the linear interpolation,

the images are growing clearer. Though this can be observed by the naked eye, this is not the ideal method to follow. In order to compare two images with two dissimilar types of interpolation, the difference in the quality will be too small, and it cannot be distinguished with the naked eye. In this situation, however, this is not the case, as we are comparing between two images: one retrieved without interpolation, and the other retrieved with linear interpolation, both with an obvious discrepancy. In the subsequent paragraphs we will explain and exemplify different types of interpolation through images. For this reason, it will prove difficult to compare these images by simply looking. We will introduce a method based on the range distance, which will lead to a conclusion regarding which interpolation method is better to utilise. This method will be explained further in this paragraph.

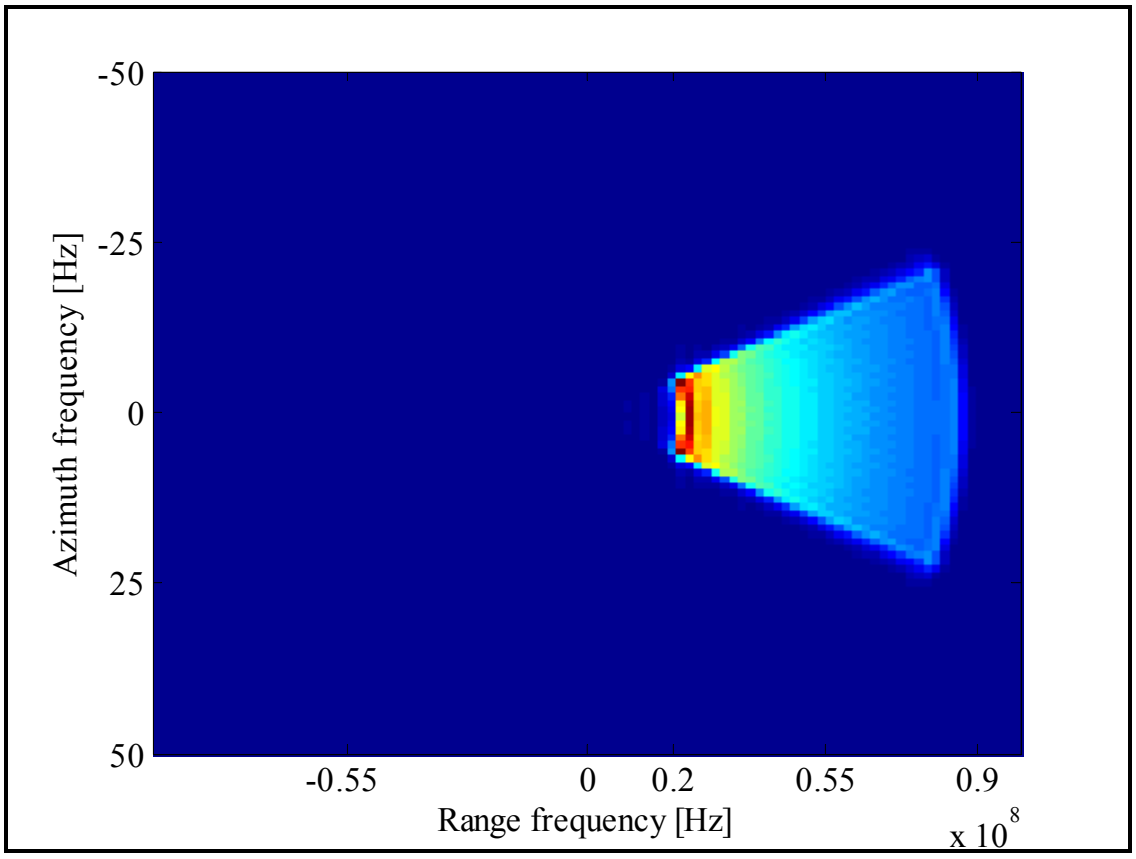

Figure 4. 4: System transfer function with linear interpolation.

Here, the process by which the SAR image matrix was captured shall be highlighted. Further, we intend to plot the image of the point target by using the linear interpolation as convolution with a triangular function. This technique was covered in detail in chapter 2, where the transfer function of the triangular function is shown in the equation 2.7, as well as the convolution formula in equation 2.6.

4.4.2 Implementation of Triangular Interpolation Technique

In the Global backprojection, this method is easy to execute due to convolvment in the range direction only, as there is no need to interpolate in the azimuth direction. The process begins in the same way as before, where the $N \times N$ empty image of size 100 is created. The transfer function of the triangular is then defined in Matlab, as previously explained, with the array $p_s = 1 - [0 : \Delta : 1]$ where Δ is equal to 0.01.

The same steps in the linear interpolation are now followed, to calculate η. However, instead of applying the formula 3.1 as used before, we convolve the received data matrix with p_s for two points, with the values η and $\eta+1$ following the formula 4.2 below:

$$h(n) = S(n) * p_s\left(\frac{\eta}{\Delta}\right) + S(n+1) * p_s\left(\frac{1-\eta}{\Delta}\right) \tag{4.2}$$

where $h(n)$ is the final image, S is the received data matrix, and p_s is the triangular function. This will lead to a different SAR image matrix which is plotted and shown with its transfer function in the frequency domain, in the figures 4.5 and 4.6 respectively.

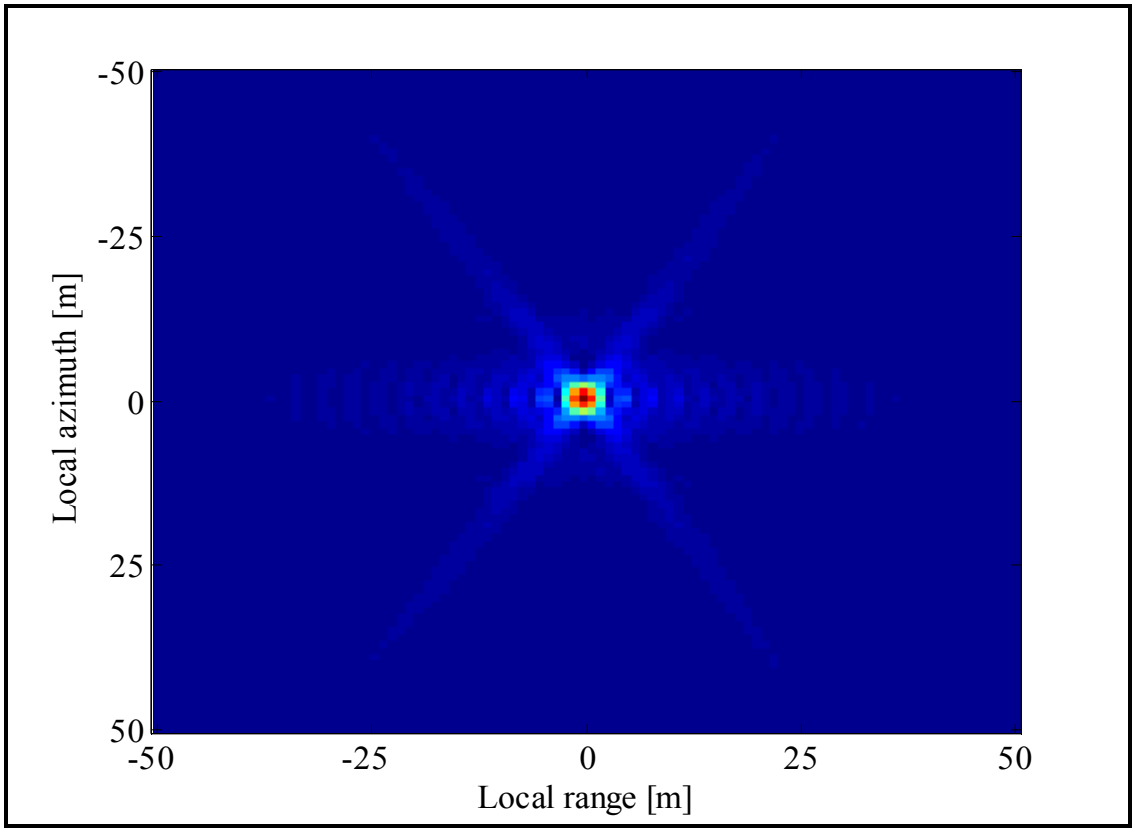

Figure 4. 5: GBP image of one point target with linear interpolation as Convolution.

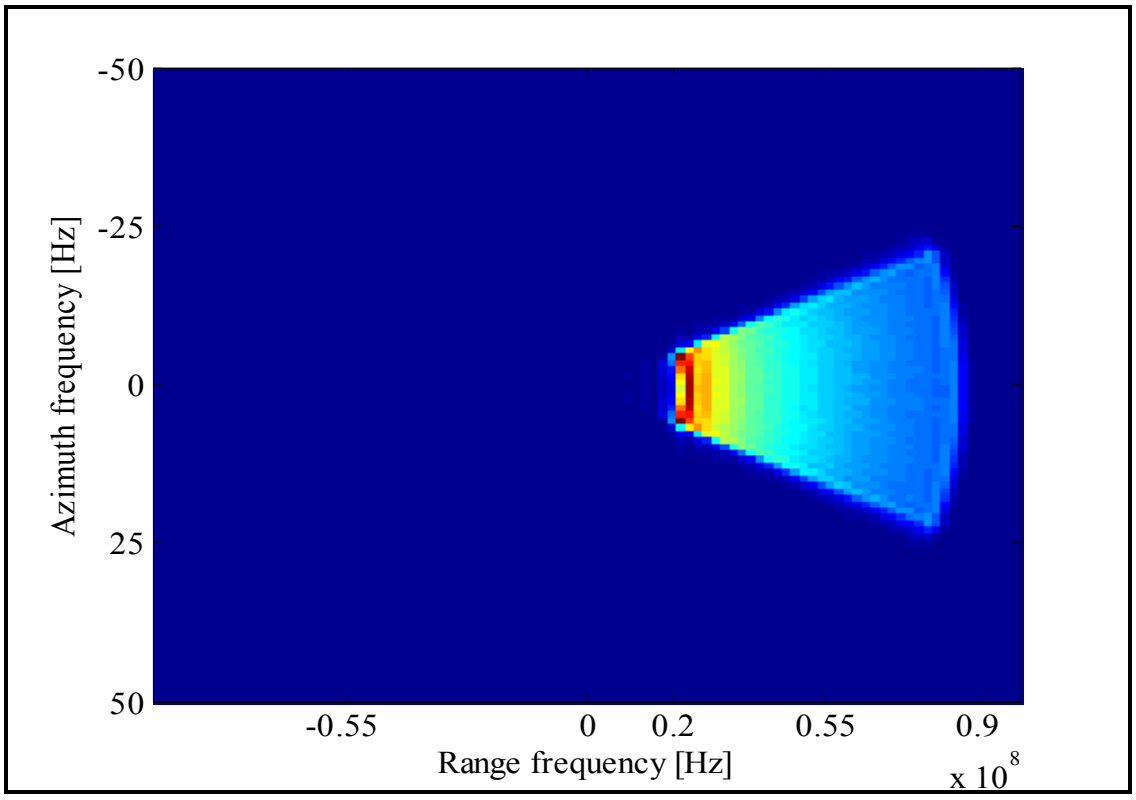

Figure 4. 6: System transfer function with linear interpolation as Convolution.

As aforementioned, it now becomes challenging to conclude which of these two interpolation methods leads to a better quality image. Consequently, we continue by examining the third type of interpolation used in this thesis work, namely, the sinc interpolation. Lastly, we shall explain the method based on the range distance. This allows for a sounder conclusion as to which interpolation method is most effective for the purpose of gaining a higher quality image.

4.4.3 Implementation of Sinc Interpolation Technique

As we concluded from the earlier Fast backprojection chapter, the sinc interpolation was the optimum interpolation to use in retrieving the best quality image. There, it was a two-directional interpolation in range, and angular, but here it will be only a 2D interpolation in range and angular.

The sinc interpolation was the hardest interpolation to actuate due to its complexity. It is, however, easier to execute in the Global backprojection, rather than in the Fast backprojection. In the Global backprojection, we only interpolate in one direction, which is not appropriate in the Fast backprojection, where it was a double sinc interpolation, and we interpolated in two directions as highlighted in Chapter 3.

Returning to the sinc interpolation, we initially create an $N \times N$ empty image, then upsample the received matrix in one direction, choosing any upsampling factor. Subsequently, we define the sinc function $\text{sinc}(\varphi)$ in Matlab as explained before.

The Global backprojection is now implemented entirely again, in order to calculate the number of delays. We run three for loops, calculating the distance between every azimuth position of the plane, and every pixel of the image, using the same formula 3.15. From the distance found, we can easily calculate the time delays and then the amount of delays. Through this, the desired delay value η is also identified. The role of the sinc interpolation is now to convolve the twelve nearest points with the sinc function, according to formula 4.3 below:

$$h(n) = \sum_{k=0}^{11} S(n+k) * \text{sinc}\left(\frac{k-\eta}{\Delta}\right) \tag{4.3}$$

After the convolution, we add the value of the SAR and save it in the SAR image matrix. After running the three for loops we will get the final SAR image data. Figures 4.7 and 4.8 indicate the plots of the point target image and its transfer function in the frequency domain.

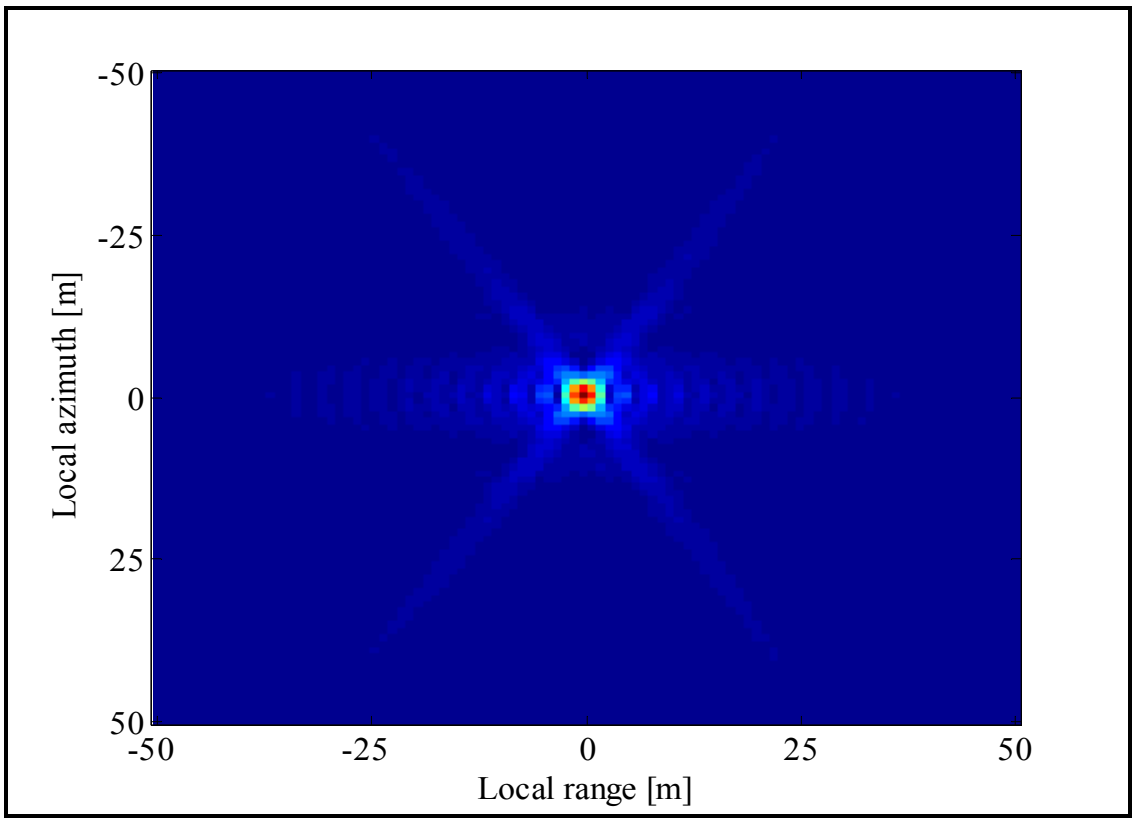

Figure 4. 7: GBP image of one point target with Sinc interpolation.

Sinc interpolation yields a very accurate and clear image since it convolves its twelve nearest points each time. It has a heavy computational load, and the processing time is longer than other interpolation. In other words, the sinc interpolation is a compromise between the image quality, the computational load and the processing time. Nevertheless, the sinc interpolation is the best interpolation to use, because the computational load and the processing time remain acceptable.

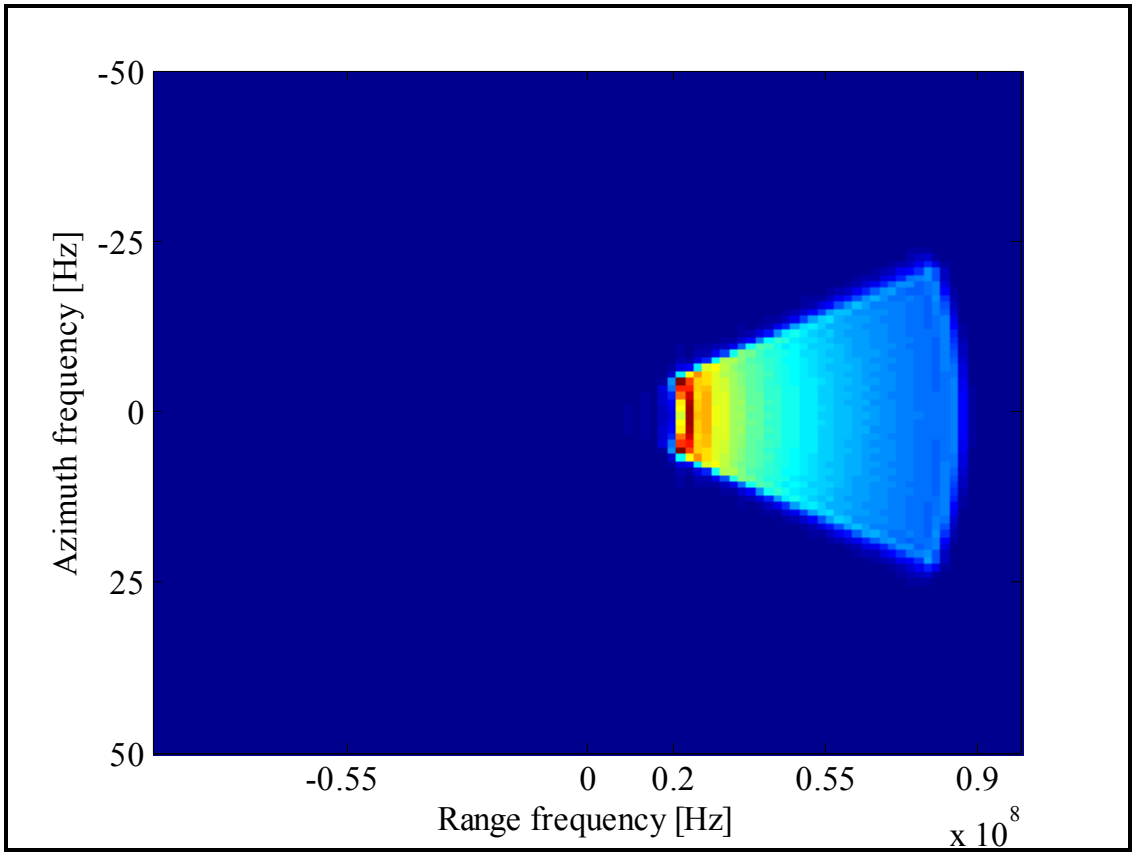

Figure 4. 8: System transfer function with Sinc interpolation.

4.4.4 Results and Conclusion

After implementing the three varieties of interpolation, it is then fitting to determine which interpolation is ideal for retrieving the optimal image. Accordingly, the range resolution calculation method outlined in section 3.3.4 in Chapter 3 is engaged. This entails that we upsample the final image matrix in the range direction by a factor of 50 or 100, and consequently plot the normalised and logarithmic SAR image data for the point target, which is located in the middle of the image. The range distance is measured at – 3dB. The smaller distance leads to an

accurate and clear image. Table 4.1 below displays the results recorded after applying this method for GBP, both with and without interpolation.

Table 4. 1: GBP results of the range resolution for linear track.

	-3dB left	-3dB right	Range resolution in m
No Interpolation	2412	2551	2.78 m
Linear Interpolation	2390	2528	2.76 m
Triangular Interpolation	2390	2528	2.76 m
Sinc Interpolation	2342	2479	2.74 m

From this table, it is established that when no interpolation is involved, the range distance of the point is wide, and this leads to an unclear image. Further, the sinc interpolation is the ultimate interpolation to use as the distance is smoother than other range distance where there exists no demarcation between the linear and the triangular interpolation.

As will be demonstrated in the comparison section, the sinc interpolation ensures the best image, but not the fastest processing time. Processing times will be tabulated, to finally assess which algorithm and interpolation to use for securing faster processing times or clearer images.

Before finishing this chapter, we repeat all the steps previously addressed, but with the consideration in mind that the flight track is not linear. The motion compensation was easily solved in the same fashion as in the Fast backprojection. We employed the range distance method and found the same distances as for a linear trajectory. Table 4.2 below reveals the results of the range resolution method. The conclusion reached is that the motion compensation problem is more easily disentangled in the time domain, rather than in the frequency domain.

Table 4. 2: GBP results of the range resolution for non-linear track.

	-3dB	-3dB	Range resolution in m
No Interpolation	2410	2549	2.78 m
Linear Interpolation	2385	2523	2.76 m
Triangular Interpolation	2385	2523	2.76 m
Sinc Interpolation	2366	2503	2.74 m

4.2 Comparative Studies

Throughout this thesis, we have studied the Fast backprojection and the Global backprojection, finding that the Fast backprojection has more advantages, being the better algorithm to use in order to balance lower computational load and retrieve good quality images in faster processing time. The image quality of the Fast backprojection is roughly the same comparing to the image quality of the Global backprojection. This is proven by calculating the range resolution, where the range in the Fast backprojection is roughly equal to range in the Global backprojection for all types of interpolation. The computational load of the Fast backprojection is reduced by a factor of $\sqrt{N}$ from that in the Global backprojection. This reduction affects the processing time.

From table 4.3 below, it is established that the processing time of the Fast backprojection is reduced by the same factor. Table 4.3 shows the processing time for an 100×100 image, implemented in both Global backprojection and Fast backprojection using the three interpolation types for upsampling factors of 4 and 8 respectively. This table indicates the processing time of these two algorithms when not implementing any interpolation techniques. As we had concluded before that the interpolation had affected the image quality, this table evidences that the interpolation techniques had a significant effect on the computational load and the processing time. Therefore, the more complex the interpolation, the more time it takes to execute, but the image quality retrieved is higher. This is the trade-off between the image quality and the processing time.

Table 4. 3: Processing time for GBP and FBP.

	No Interpolation	Linear Interpolation	Triangular Interpolat.	Sinc Interpolat.
GBP/4	8.127 s	15.438 s	20.120 s	52.172 s
FBP/4	3.917 s	11.706 s	12.682 s	19.432 s
GBP/8	8.122 s	21.223 s	24.966 s	57.298 s
FBP/8	3.929 s	17.725 s	19.039 s	35.761 s

The processing time in the Fast backprojection was the fastest in all cases, for different kinds of interpolation and for different upsampling rates. For example, comparing the processing time for an 100×100 image with the sinc interpolaton and an upsampling factor of 4, we conclude that the processing time for FBP was reduced by 32.74 seconds from the processing time for GBP. It is a huge improvment in processing time. We conclude that when the upsampling factor increases, the processing time increases. This is due to the increase in the number of operation.

CHAPTER 5: CONCLUSION AND FUTURE WORK

In conclusion, FBP retains the advantages of GBP as perfect motion compensation, an unlimited scene sizes, wide bandwidth and ability to handle long integration angles. However, the Fast backprojection has more advantages as we can conclude that the computational load in the Fast backprojection algorithm is improved by a factor of $\sqrt{N}$ since the Global backprojection has a computational load of N^3. So, the FBP processing time for retrieving a high quality image is then faster than GBP. The image quality retrieved by FBP is roughly the same compared to the quality retrieved by GBP. Experiments on FBP and GBP with different interpolation thecniques show that the sinc interpolation implemented is the best interpolation used in term of image quality.

This thesis has presented a study for FBP for image retrieval of the Synthetic Aperture Radar (SAR). We have examined various types of interpolation to enhance potential image quality, as well as showcasing a comparison between FBP and GBP in term of processing time and the image quality. The computational load and processing time are highlighted, and we study the difference between these algorithms for different types of interpolation techniques. Time constraints restricted the completion of further investigations. Consequently, this thesis could be enhanced in the future by studying the Fast Factorized Backprojection (FFBP), which has a computational load of $N/\log(N)$ [4]. This thesis work can also be improved through study of the frequency-domain algorithms such as Range Migration Algorithm (RMA) and the Range Doppler Algorithm (RDA). A further comparison could be presented based on this data, in relation to computational load, processing time and image quality between all the time-domain algorithms. A comparison could be also made between all frequency-domain algorithms. Further investigation could include studying and comapring between the time-domain algotithms and the frequency-domain algorithms.

REFERENCES

[1] W.G. Carrara, R.S. Goodman, and R.M. Majewski, *Spotlight Synthetic Aperture Radar, Signal Processing Algorithms*,Boston: Artech House, 1995,iSBN: 0-89006-728-7.

[2] M. Soumekh, *Synthetic Aperture Radar Signal Processing*. John Wiley & Sons inc. USA, 1999, iSBN:0-471-29706-2

[3] Sandia National Laboratories http://www.sandia.gov/radar/whatis.html.

[4] V. T. Vu, T. K. Sjogren, M. I. Pettersson, "*A comparison between Fast Factorized Backprojection and the Frequency-Domain Algorithms in UWB Low Frequency SAR*," IEEE Geosci. Remote Sensing Symp., July 2008.

[5] A. F. Yegulalp, "*Fast backprojection algorithm for synthetic aperture radar*," IEEE Radar Conf., April 1999.

[6] A. Åhlander, H. Hellsten, K. Lind, J. Lindgren, and B. Svensson, " *Architectural Challenges in Memory-Intensive, Real-Time Image Forming,* " IEEE Parallel Processing Conf., 2007.

[7] T. K. Sjogren, V. T. Vu, M. I. Pettersson, " *A Comparative Study of the Polar Version with the Subimage Version of Fast Factorized Backprojection in UWB SAR*," Internl. Radar Conf., 2008.

[8] W. S. George, *Introduction to Airborne Radar* second edition. Scitech publishing.inc USA, 1998, iSBN:1891121014.

[9] M.I. Pettersson, V. Zetterberg, I. Claesson, "*Detection and imaging of moving targets in wide band SAS using fast time backprojection combined with space time processing,* " Blekinge Department of signal processing, 2005.

[10] J. G. Proakis, D. G. Manokalis, *Digital Signal Processing principles algorithms and applications*, Third edition, Prentice-Hall International, inc. USA, 1996, iSBN:0-13-394289-9.

[11] G.B.Thomas, M.D.weir, J.Hass, F.R.Giordano, *Thomas Calculus*, Eleventh edition, Addison-Wisley, Prentice-Hall International, inc. USA, 2005, iSBN -10: 0-321-18558-7.